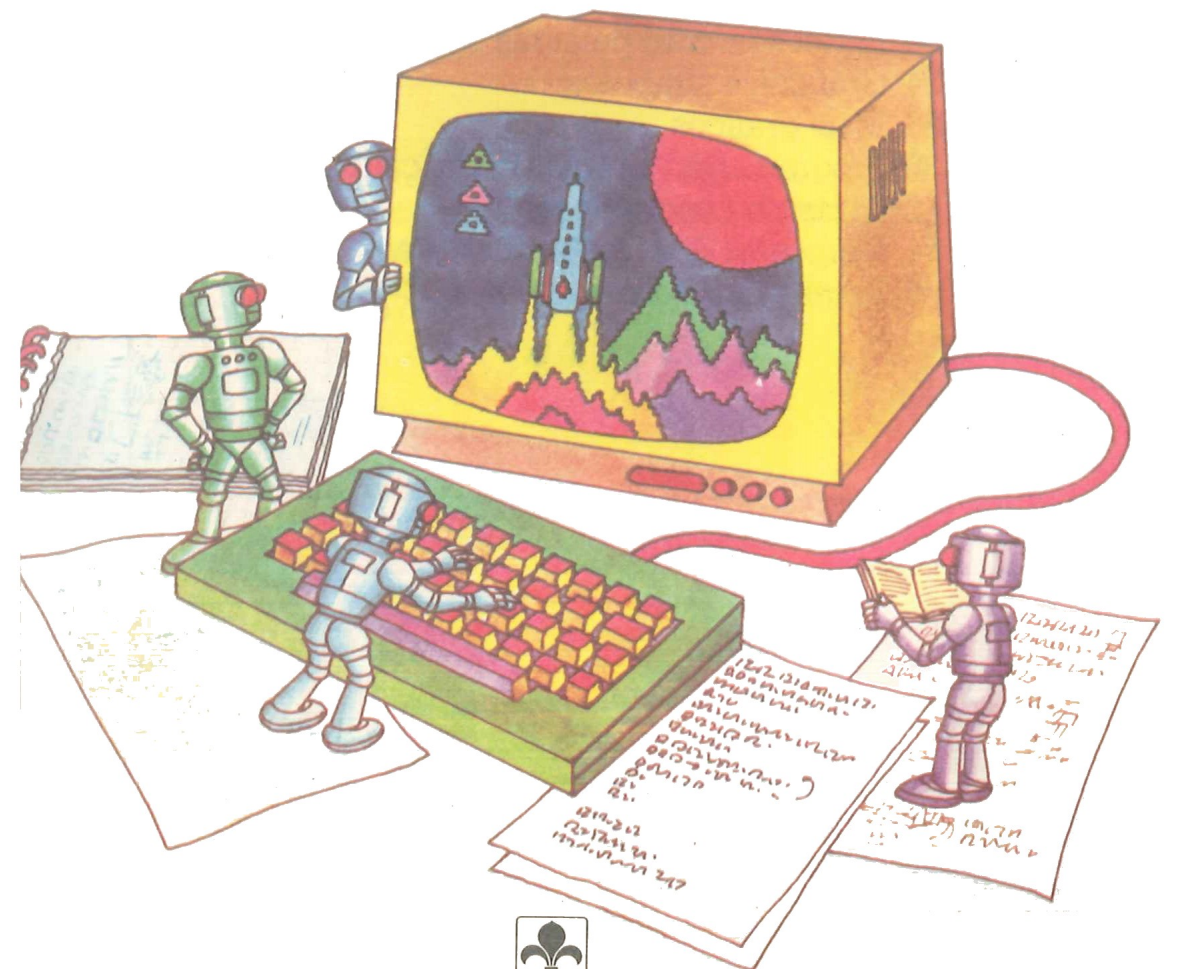
GUIA PRÁTICO
DE PROGRAMAÇÃO EM
BASIC
2ª EDIÇÃO
CD 60
SONY
RETURN
SHIFT
Iniciação à
programação
EDITORA LUTÉCIA

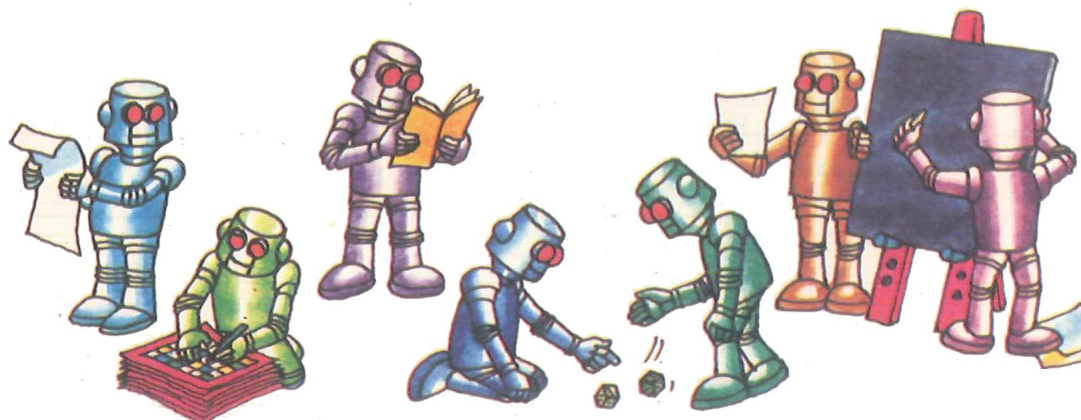
GUIA PRÁTICO DE PROGRAMAÇÃO EM BASIC

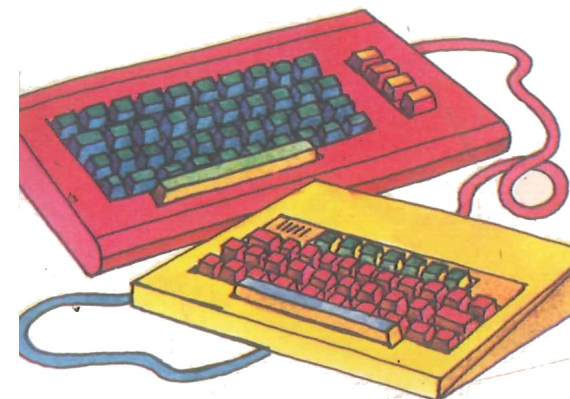
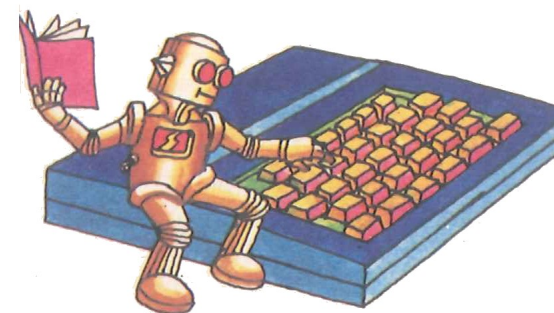
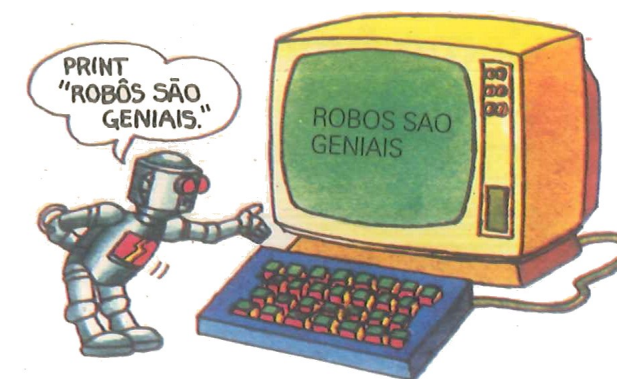
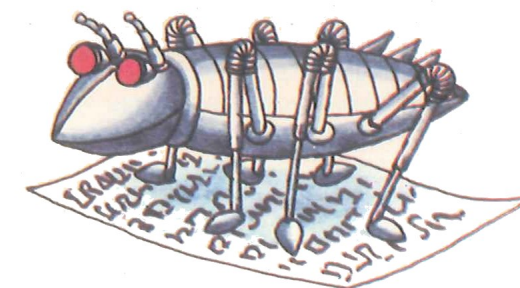
Brian Reffin Smith

Diagramação de Kim Blundell
Ilustrações de Graham Round e Martin Newton

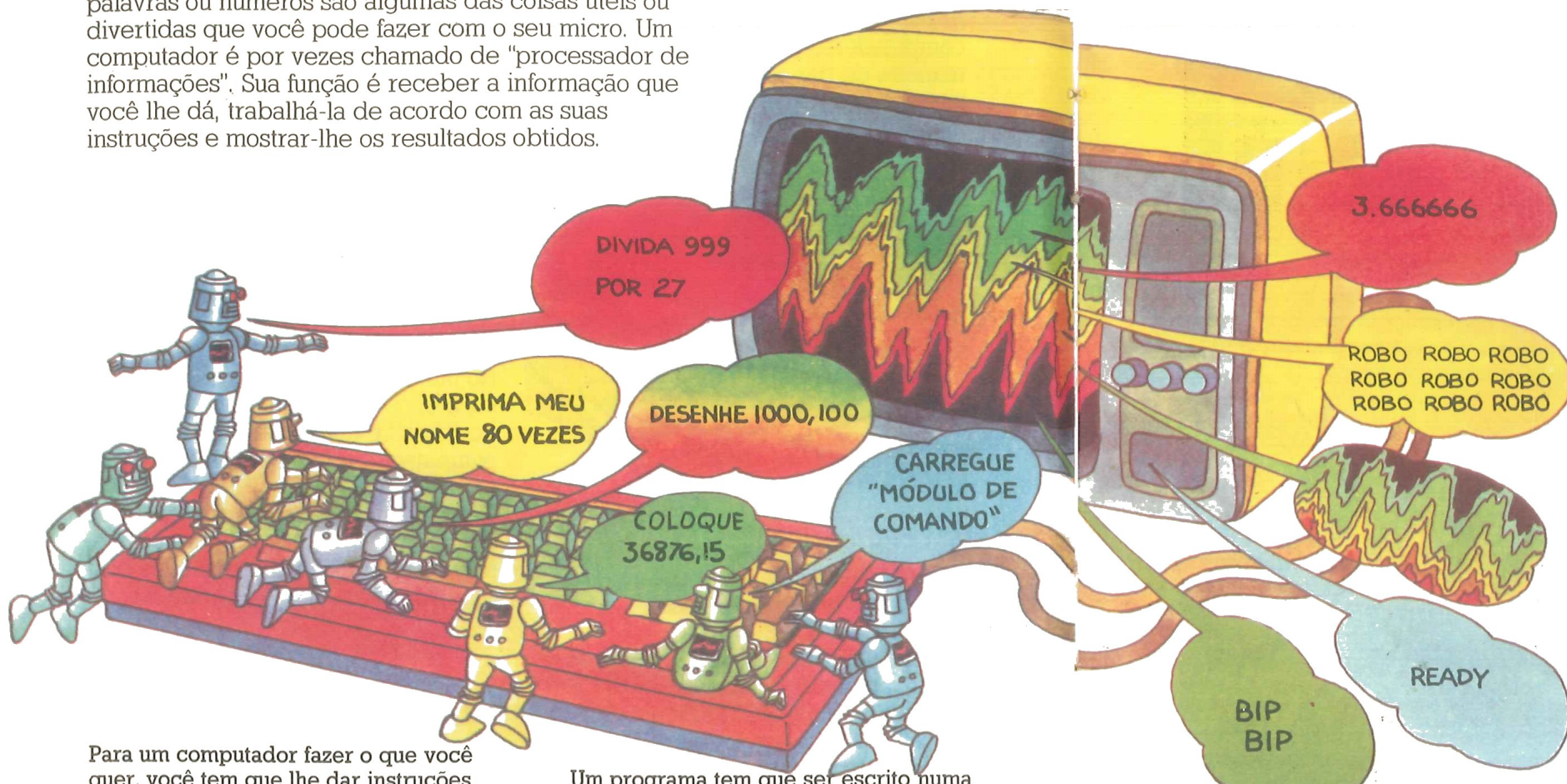
Tradução e Adaptação de Sérgio Machado

2ª Edição

LUTÉCIA

Sumário

Introdução

Este livro é um guia para principiantes em BASIC. BASIC é a linguagem mais usada nos computadores pessoais. É uma maneira de dar ao computador instruções que ele entende. Você não precisa de um computador para

usar este guia, apesar de que ajudaria bastante a entender os programas se você pudesse experimentá-los.
Marcas diferentes de computadores usam versões ligeiramente diferentes de BASIC. Praticamente todos os termos usados neste livro funcionarão na maioria dos microcomputadores e as poucas

exceções estão claramente indicadas.
Começaremos dando algumas linhas gerais sobre programação. Depois, à medida que você for lendo este livro, as principais instruções de BASIC serão apresentadas uma a uma, com pequenos programas para mostrar como elas funcionam.

Para você adquirir alguma prática em programação, este livro apresenta alguns "quebra-cabeças" para você resolver e sugestões para criar ou modificar programas. (As soluções estão nas páginas 44 e 45.) No fim deste livro (páginas 46 e 47) você encontra uma lista de termos de BASIC e outras palavras de computação com uma breve explicação.
Damos também em poucas linhas algumas "dicas" para programação e uma lista dos erros mais comuns com sugestões para você encontrá-los.

Se você tiver um micro, experimente os programas deste livro, e para saber mais como ele funciona leia o manual do fabricante que o acompanha. Pode ser que algumas das regras indicadas aqui não sejam necessárias ao seu micro. A melhor maneira de aprender BASIC é experimentando o máximo de programas possível, seja de livros seja de revistas especializadas, alterando-os um pouco para ver o que acontece. Logo, logo você estará fazendo os seus próprios programas em BASIC.

Como funciona um computador

Você pode fazer um monte de coisas com um computador. Desenhar, fazer gráficos, jogos e processar palavras ou números são algumas das coisas úteis ou divertidas que você pode fazer com o seu micro. Um computador é por vezes chamado de "processador de informações". Sua função é receber a informação que você lhe dá, trabalhá-la de acordo com as suas instruções e mostrar-lhe os resultados obtidos.

Para um computador fazer o que você quer, você tem que lhe dar instruções muito precisas. Uma relação de instruções para um computador é chamada de *programa* e as informações que você dá são chamadas de *dados.*

Um programa tem que ser escrito numa linguagem que o computador possa entender, tal como BASIC, e o programador tem que seguir todas as regras desta linguagem.

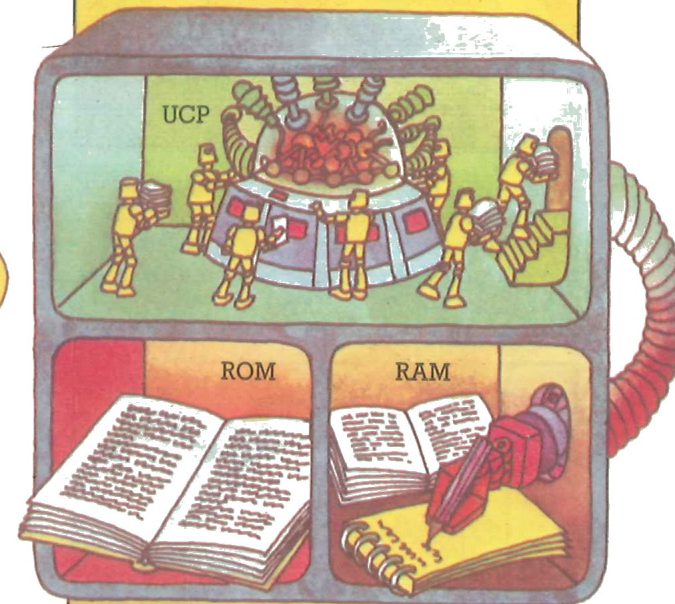
Microcomputadores

A maioria dos micros consiste em um teclado que você liga num aparelho de TV. As instruções e informações são dadas através do teclado e tudo o que você informar junto com os respectivos resultados aparecerão na tela da televisão.

Alguns micros têm sua própria tela, algumas pequenas como a de uma calculadora portátil. Outros têm uma tela especial chamada *monitor.* O monitor é uma televisão que não pode receber os canais das estações de TV. O teclado de um micro parece uma máquina de escrever com algumas teclas extras. Em alguns micros cada tecla representa uma instrução de BASIC, de tal forma que você não precisa escrever a instrução letra por letra.

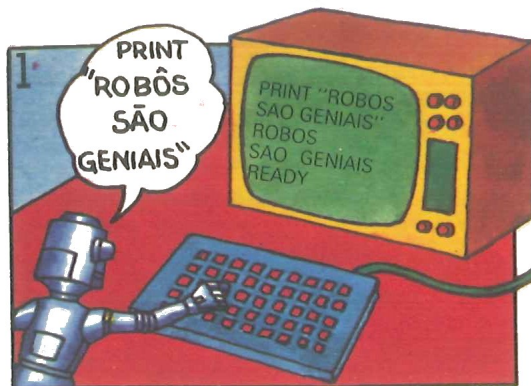
A TV é a maneira mais usual de apresentar as informações de um micro. Você pode, entretanto, receber estas informações em papel, usando uma impressora. Isto é muito útil, já que as informações do micro na tela de TV são perdidas quando você desliga o computador.

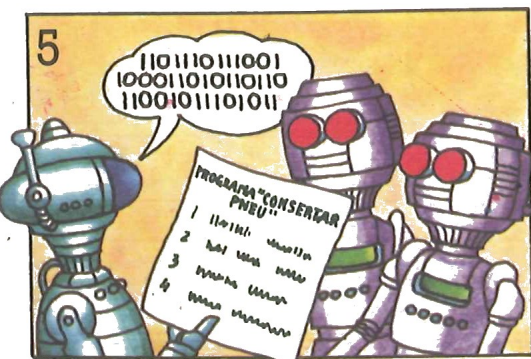
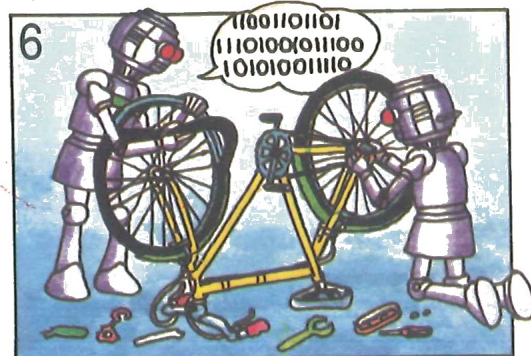
Dentro de um micro

Um microcomputador é composto de duas partes: a unidade central de processamento (UCP), onde todo o trabalho é realizado e a memória, onde os programas e os dados são guardados.

Na verdade, o computador tem duas memórias. Uma, chamada ROM (Read Only Memory = Memória Apenas para Ler), contém um programa permanente. A outra, chamada RAM (Random Access Memory = Memória de Acesso Aleatório), é uma memória vazia, onde ficam os seus programas e os dados e é apagada quando você desliga o micro.

Impressora

Uma outra maneira de guardar informações é num gravador cassete. Programas e dados podem ser gravados em cassete, e quando você precisar deles basta "carregá-los" no micro.

Dando instruções ao computador

Para que o computador faça alguma coisa, você tem que usar instruções que ele entenda. A instrução pode ser um comando direto que ele executa imediatamente ou pode ser um programa de instruções que, guardadas na memória, só são executadas quando você mandar.

As instruções num programa devem ser cuidadosamente formuladas. O computador vai tentar executar as suas instruções mesmo que elas estejam erradas. O computador não entende as instruções escritas em nossa língua, de forma que você tem que escrevê-las em uma das diversas linguagens de computador.
Algumas dessas linguagens são descritas na página seguinte.

Todo o trabalho realizado dentro do computador é feito através de um código de pequenos impulsos elétricos. As instruções são traduzidas para este código chamado linguagem de máquina por um programa especial chamado de interpretador.

Cada informação em linguagem de máquina é representada por uma seqüência de impulsos. A linguagem de máquina pode ser escrita usando-se o algarismo *1* para representar um impulso e o *0* como ausência de impulso.

Linguagens de computador

Você poderia escrever programas em linguagem de máquina, mas isso iria ser muito difícil. No entanto, há linguagens especiais de computador, chamadas de alto nível, as quais o computador pode traduzir para o seu próprio código.

Existem centenas de linguagens de alto nível diferentes, algumas delas desenvolvidas para um certo tipo especial de trabalho. BASIC é uma das linguagens mais difundidas. Suas letras significam, em inglês BEGINNER'S ALL-PURPOSE SYMBOLIC INSTRUCTION CODE, ou seja, Linguagem Geral para Principiantes. Mas BASIC não é somente usada por principiantes. Abaixo, nós temos alguns exemplos de diferentes linguagens.

Programa nomes

```
10 PRINT "QUAL E
   O SEU NOME?"
20 INPUT N$
30 IF N$ = "JOAO" THEN PRINT
   "OI, TUDO BEM?"
40 IF N$ ="JORGE" THEN PRINT
   "VA EMBORA"
50 END
```

Programa dinheiro

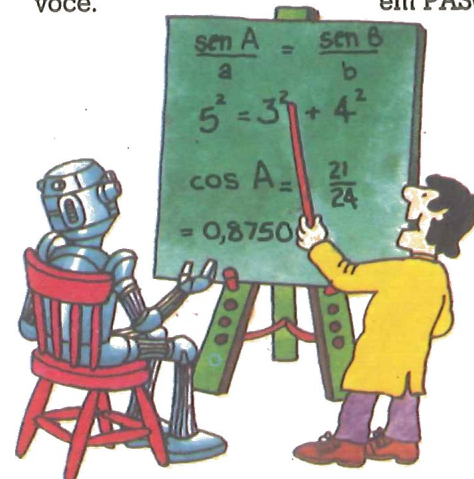
```
PROCEDURE DINHEIRO;
VAR MES: INTEGER
BEGIN (* CALCULO *)
FOR MES = 1 TO TOTALMESES DO
TOTALDEVIDO: =
(VALORDEVIDO-JAPAGO)*
(1+JUROS)
END; (*DINHEIRO*)
```

Programa geografia

```
T: QUAL E A CAPITAL DA FINLANDIA
* INPUT A: $A$
T (LEN(A$)=0) : VAMOS, DE UM CHUTE
JC: @ A
M: COPE! STOCK! OSLO! GOTHEN! LAP
TY: ESTA QUENTE, MAS NAO ESTA CERTO —
    TENTE DE NOVO
JY: @ A
M: HELSINKI! HELSING! HELSINQUE!
TY: MUITO BEM — E HELSINQUE
```

Este é um pequeno programa em BASIC. A linha 10 manda o computador imprimir "Qual é o seu nome" na tela. Então o computador guarda o seu nome na memória, e se ele for João ou Jorge imprimirá uma mensagem na tela para você.

Este programa foi escrito em PASCAL, uma linguagem cujo nome é uma homenagem ao famoso matemático francês. É uma parte de um programa para calcular juros sobre dinheiro. Muita gente acha mais fácil escrever bons programas em PASCAL do que em BASIC.

Esta é uma linguagem chamada PILOT. É muito usada para programas que ajudam as pessoas a aprender novos assuntos. Nesta linguagem, o computador é capaz de reconhecer respostas mesmo que elas não estejam absolutamente certas.

11. Bb3, Ce5
12. 0-0-0, Cc4
13. Bxc4, Txc4
14. h5, Cxh5

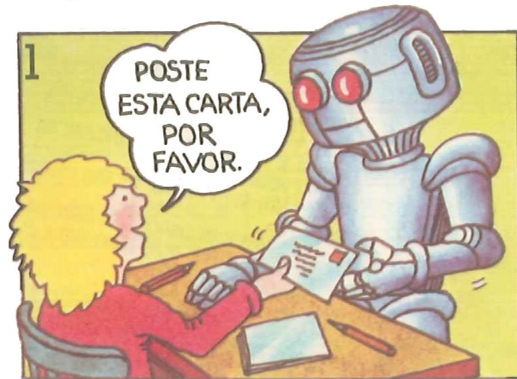
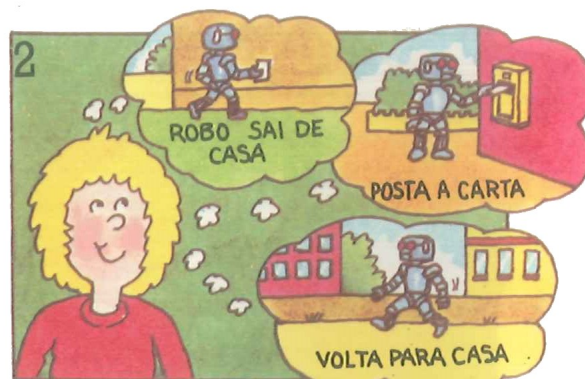
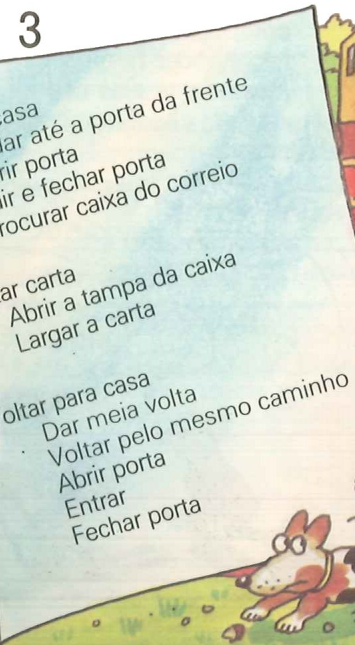
À primeira vista, linguagens de computador parecem ser muito estranhas e complexas, mas assim são outras línguas como o finlandês mostrado no quadro ao lado. Existe uma série de outros assuntos, também, em que linguagens especiais são usadas.

Em matemática, por exemplo, uma notação especial é usada para escrever idéias e fórmulas para as quais seriam necessários montes de palavras. A mesma coisa acontece na música e no xadrez.

* Uns 15 abaixo de zero.

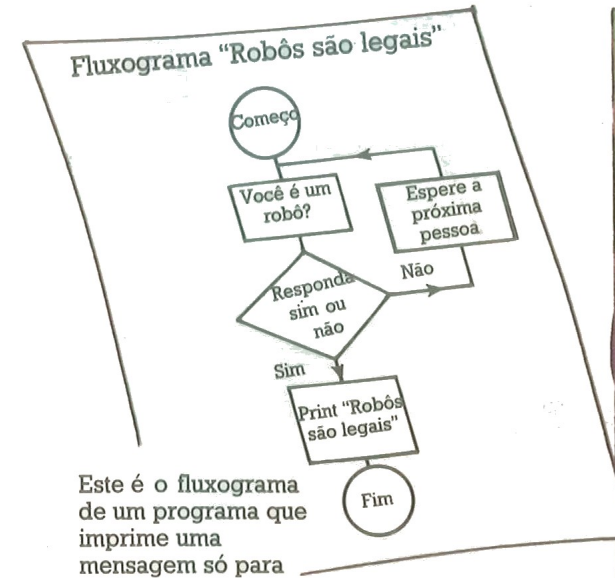
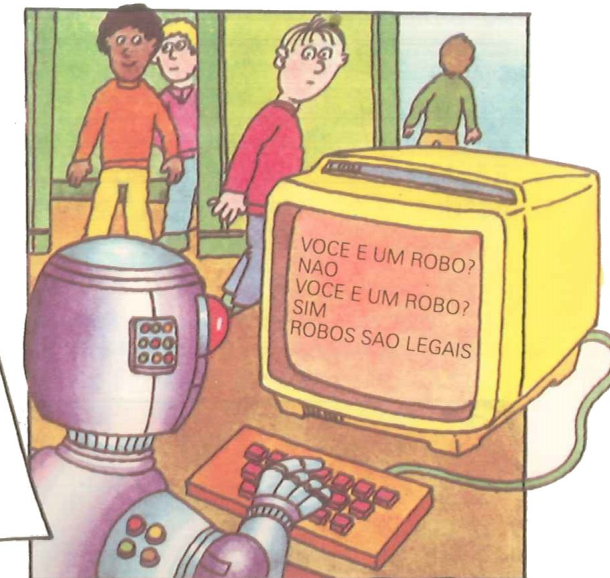
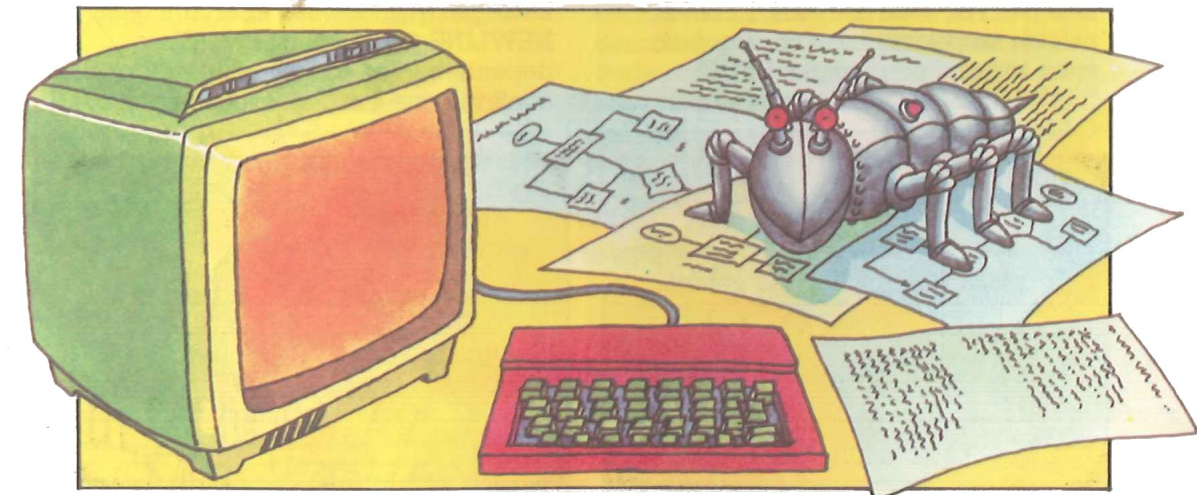
Programando

Um programa é como as regras de um jogo, ou uma receita de cozinha. Se houver um erro nas regras ou na receita, você não vai poder jogar direito nem fazer um bolo gostoso. Da mesma maneira, os resultados que você obtém do computador dependem das instruções que você deu. Para programar, você precisa estudar o que quer fazer com muito cuidado e prever os passos principais necessários para alcançar o resultado desejado.

Programa carta

Imagine escrever um programa para ensinar um robô a postar uma carta. Uma instrução simples, como mostrada acima, seria muito difícil para o cérebro do robô entender. Você tem que prever tudo que o robô precisa fazer para postar a carta. Seu computador precisa de instruções que digam o que fazer a cada etapa.

Para programar, você deve dividir cada instrução de cada etapa em ordens bem simples, que possam ser traduzidas para a língua do robô.

O robô vai tentar executar as suas instruções, mesmo que elas estejam erradas ou incompletas. Erros num programa são chamados de "grilos" e podem, às vezes, causar resultados estranhos.

Fluxogramas

Quando se vai programar, às vezes é útil fazer um diagrama, ou fluxograma, indicando as etapas principais necessárias à resolução do problema. Assim, é possível situar cada etapa do programa.

Este é o fluxograma de um programa que imprime uma mensagem só para robôs.

As diversas etapas estão indicadas por símbolos diferentes. Começo e fim estão marcados por um círculo. As ordens de execução estão em retângulos, e as etapas de decisão (onde o computador age de acordo com a resposta recebida) ficam em losangos. As linhas e flechas mostram o itinerário que o computador pode seguir.

Depois de estudar todos os detalhes do programa, você pode convertê-lo em BASIC e testá-lo no computador. Entretanto, é possível que ele não venha a funcionar na primeira tentativa, já que pode haver algum "grilo".

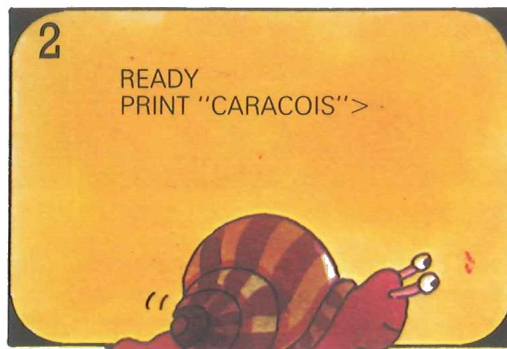
Podem ser erros de datilografia feitos na hora de entrar com o programa na máquina, ou erros de lógica. Antes de fazer o programa funcionar, temos de detectar e eliminar os "grilos"*. Às vezes, um "grilo" pode fazer o programa produzir um resultado preferível e acaba sendo útil.

* Algumas "dicas" para encontrar "grilos" estão nas págs. 42/43.

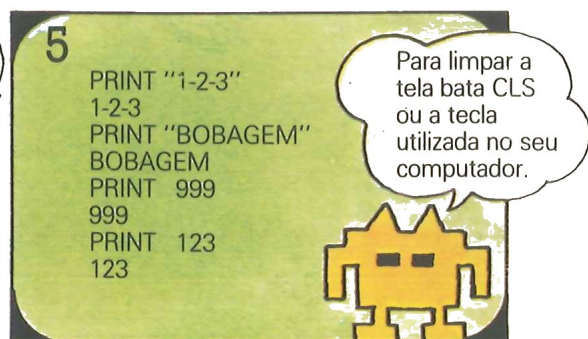
Primeiros passos em BASIC

A maioria dos termos usados em BASIC são ingleses. Assim "PRINT" significa "imprimir na tela", "RUN" é a ordem para executar um programa, "INPUT" permite dar uma informação ao computador. Nestas duas páginas, explicamos como utilizar a instrução PRINT.

A maioria dos computadores pessoais possui um interpretador de BASIC residente. Quando são ligados, já estão prontos para serem programados em BASIC*.

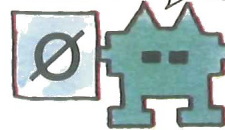
Quando você liga o micro, algumas palavras aparecem na tela, junto com um pequeno símbolo chamado cursor. O cursor mostra onde a próxima letra que você bater vai aparecer.

Para fazer o computador imprimir na tela uma palavra, você usa PRINT com a palavra ou palavras que você quiser, entre aspas. Por exemplo: PRINT "CARACOIS" manda imprimir a palavra CARACOIS na tela.

O computador só vai executar a sua instrução quando você apertar a tecla NEWLINE (RETURN ou ENTER — dependendo da marca do aparelho) que confirma que a ordem está completa.

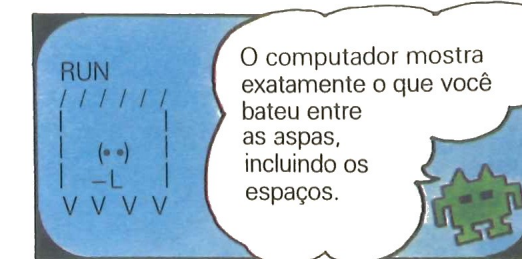
O computador irá mostrar na tela tudo que você bateu entre as aspas. Podem ser letras, números, palavras ou símbolos. Observe que ele não mostra as aspas.

Para imprimir apenas números, você não precisa de aspas. Para limpar a tela, basta bater CLS na maioria dos micros (confira no manual, se você tiver um computador).

* Alguns computadores necessitam de um programa especial, carregado através de uma fita cassete, para entenderem BASIC.

Um programa em BASIC

Num programa, cada linha começa com um número. Isto indica ao computador que as instruções devem ser guardadas na memória para só serem executadas quando você der o sinal. Na página oposta, as instruções ao computador não tinham números, e portanto ele as executava imediatamente. Aqui, apresentamos um programa que permite ao computador imprimir na tela símbolos que formam um rosto.

```
10 PRINT "/////"
20 PRINT "I     I"
30 PRINT "I (..) I"
40 PRINT "I  -L  I"
50 PRINT "V V V V"
60 END
```

As linhas de programas são geralmente numeradas de 10 em 10, para que seja possível acrescentar novas instruções sem ter que renumerar todo o programa.

Quando você entra com um programa, aperte a tecla NEWLINE (ou aquela correspondente no seu micro) no fim de cada linha. As instruções são mostradas na tela, mas o computador não as executará até que você dê a ordem RUN. Cuidado para não confundir a letra O com o número 0, pois isto causará um erro. Use a tecla RUBOUT ou DELETE para corrigir erros de datilografia.

Quando você tiver entrado com todas as instruções, confira com cuidado para verificar se não há nenhum erro. Para mandar o computador executar o programa, bata RUN seguido de NEWLINE.

Se o programa não funcionar ou se o resultado não for o esperado, é necessário listar o programa na tela para descobrir o erro. Neste caso bata LIST. Pode ser que o computador imprima uma mensagem indicando o erro.

O computador indicará a maioria dos erros através de uma mensagem. Cada mensagem de erro está explicada no manual do computador. A maneira mais fácil de corrigir um erro é rebatendo a linha inteira novamente. O computador substituirá a linha errada pela nova. Para se eliminar uma linha de um programa basta bater o número da linha seguido de NEWLINE.

Cada computador tem a sua própria maneira de corrigir ou alterar parte de linhas usando as funções EDIT ou COPY. Isto está detalhado no manual do computador.

Quebra-cabeça — Tente mudar o programa para modificar a face da figura desenhada.

Entrando com os dados

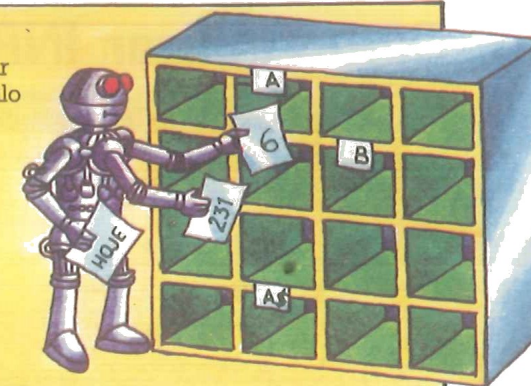
Para que o computador execute tarefas mais interessantes do que simplesmente mostrar símbolos na tela, é necessário fornecer-lhe informações ou dados que ele possa utilizar.
O computador guarda estas informações em sua memória até que você lhe dê a ordem para usá-las.

Ao introduzir um dado na memória do computador, você tem que atribuir-lhe um rótulo para que possa encontrá-lo novamente. Você pode usar letras do alfabeto como rótulos. Para rotular um espaço de memória e colocar um número nele, use a instrução LET como está mostrado acima. Um espaço de memória rotulado é chamado de variável porque pode guardar dados diferentes em momentos diferentes do programa.

Você deve usar um tipo diferente de rótulo para guardar letras e símbolos. O conjunto de uma ou mais letras ou símbolos é chamado de "string". Os rótulos de strings são letras do alfabeto seguidas do sinal $.

Você coloca o string no espaço de memória usando LET, da mesma forma que uma variável numérica, mas as letras e símbolos devem ser escritos entre aspas, como no programa acima.

Para mostrar a informação na tela, utilize a instrução PRINT seguida do nome da variável, por exemplo PRINT A $. Este pequeno programa permite imprimir na tela o valor das variáveis B, D$ e L$.

Você pode rodar o programa quantas vezes quiser, que o computador apresentará sempre o mesmo resultado. Os valores das variáveis serão os mesmos até você mudá-los.

Uma outra maneira de guardar informações é através das instruções READ e DATA, como mostrado acima. A instrução READ indica ao computador o rótulo para os espaços de memória e a instrução DATA contém as informações.

Quando se roda o programa, o computador coloca cada dado no espaço de memória correspondente. Cada dado deve ser separado do próximo por meio de uma vírgula para que o computador possa identificá-los.*

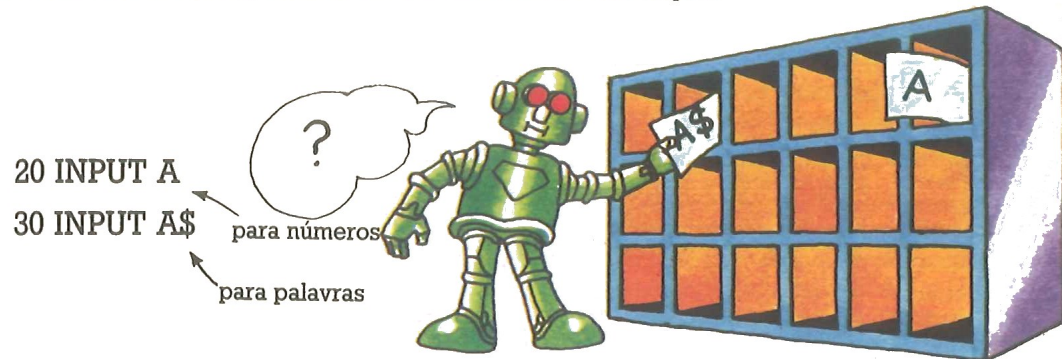
Aqui estão dois programas. Um, usando READ e DATA e o outro, usando LET para guardar a informação na memória do computador.

As variáveis são espaços rotulados na memória onde a informação é guardada e cujo conteúdo pode ser alterado ao longo do programa. Uma variável contendo números chama-se variável numérica e uma variável que contém letras e símbolos é uma variável alfanumérica.

Alguns computadores podem usar palavras como rótulos de variáveis, com exceção das palavras que são instruções em BASIC para não confundir o computador.

*Você não pode usar este método no TK 82/83/85 ou CP200.

Como utilizar INPUT

Uma outra maneira de fornecer informações ao computador é através da instrução INPUT. Ela permite introduzir dados no decorrer do programa e alterá-los a cada execução.

Você utiliza INPUT com um rótulo A se o dado for numérico, e A$ se for alfanumérico. Quando o computador encontra a instrução INPUT no decorrer de um programa, ele coloca o rótulo num espaço de memória e pergunta qual é o dado, colocando um ponto de interrogação na tela. O dado fornecido através do teclado é guardado na memória, prosseguindo então a execução do programa.

A figura 2 mostra o que acontece quando você roda este programa. Ao encontrar INPUT na linha 10, o computador imprime na tela um ponto de interrogação e espera que seja indicado o valor de G. Depois ele imprime um novo ponto de interrogação para o INPUT da linha 20. Desta vez você deve entrar com palavras ou símbolos, já que o rótulo B$ indica ao computador para esperar um string.

4

```
RUN
QUAL E O SEU NOME
? ROBO ENFERRUJADO
QUANTOS ANOS VOCE TEM
? 77
ROBO ENFERRUJADO
TEM
77
```

Se você tiver um computador, tente copiar este programa e aperte RUN para fazê-lo funcionar. Quando o computador lhe perguntar, bata o seu nome e idade, como mostrado acima. Experimente várias vezes, mudando o nome e a idade e apertando RUN para começar o programa de novo. O computador mostrará sempre o que você colocou em N$ e A.

Um pouco de poesia

Agora você já sabe BASIC suficientemente para escrever um poema no computador. Aqui, apresentamos um programa que utiliza as instruções PRINT e INPUT.

```
10 PRINT "QUAL E O SEU NOME"
20 INPUT N$
30 PRINT "UM POEMA DE"
40 PRINT N$        — Esta linha imprime o seu nome
50 PRINT "BATA UMA PALAVRA"
60 PRINT "QUE RIME COM COMPUTADOR"
70 INPUT A$
80 PRINT "AQUI ESTA O POEMA"
90 PRINT "QUEM TEM MEDO DE
         COMPUTADOR?"
100 PRINT "MEDO NAO TENHO, TENHO"
110 PRINT A$       — Esta linha imprime a sua palavra
120 END
```

Bata RUN para recomeçar com uma outra palavra.

Ao executar o programa, o computador pergunta o seu nome, guarda a resposta em N$ e depois a imprime na tela na linha 40. Ele guarda a palavra que você escolheu em A$ e a escreve na linha 110, como se estivesse integrada ao poema. Se você tiver um computador, teste este programa várias vezes, entrando com palavras diferentes na linha 70.

Quebra-cabeça

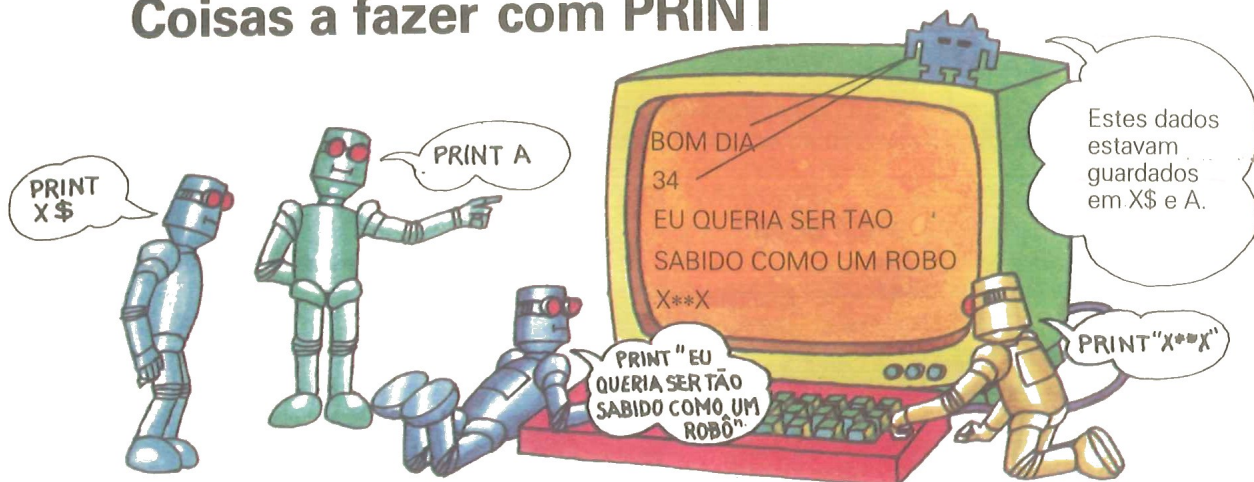
Escreva um programa no qual o computador pergunta o seu nome, depois diz bom dia, seguido de seu nome e de uma mensagem para você.

(solução na página 44)

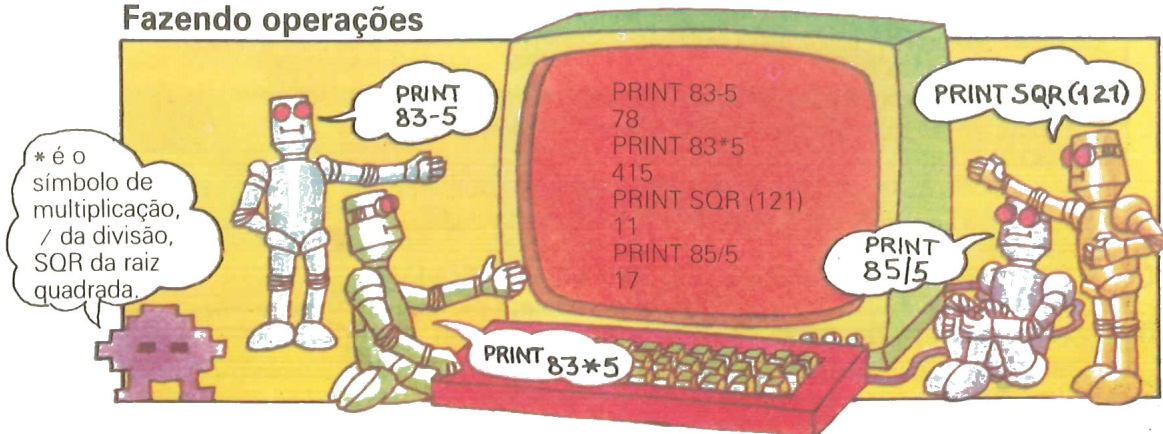
O que é necessário antes de entrar com um programa

1. Antes de entrar com um novo programa, bata NEW. Assim, você limpa da memória qualquer programa antigo e suas variáveis.
2. Quando você estiver entrando com o programa, não se esqueça de bater NEWLINE (ou a tecla correspondente) ao fim de cada linha de instrução.
3. Depois de entrar com o programa, verifique cada linha para certificar-se de que não há nenhum erro de datilografia ou linhas omitidas.
4. Você pode bater CLS (ou a instrução correspondente no seu computador) para limpar a tela. Bata RUN para começar o programa.
5. Para fazer o programa reaparecer, seja para verificá-lo ou modificá-lo, bata LIST. Para mostrar uma linha em particular, você pode bater LIST seguido do número da linha. Verifique este comando, porque ele varia de computador para computador.
6. Para interromper o programa durante a execução, bata BREAK ou ESCAPE. Aqui, também, verifique o seu manual. Em alguns computadores, ESCAPE apaga o programa da memória. Para recomeçar, bata RUN.

Ver nas páginas 42/43 como evitar os "grilos."

Coisas a fazer com PRINT

Até agora você aprender a utilizar PRINT para mostrar números e letras na tela e para ver o conteúdo de variáveis. Vamos aprender agora a usar as vírgulas e o ponto e vírgula para situar o que aparece na tela. Com PRINT podemos também fazer contas. Na página oposta damos outros exemplos de utilização das variáveis.

Estas linhas mostram como usar vírgula e ponto e vírgula para indicar ao computador onde imprimir o dado. Uma vírgula indica que deve ser deixado um espaço; um ponto e vírgula significa que é necessário imprimir junto. A figura acima mostra como as linhas seriam apresentadas na tela. A palavra PRINT sozinha faz o computador saltar um linha.

Fazendo operações

Você sabe agora como utilizar PRINT. Para a soma e a subtração use os sinais + e − habituais; use * para a multiplicação e / para a divisão.

O computador pode fazer cálculos matemáticos mais complexos, como seno, co-seno, raiz quadrada etc.

Para saber mais sobre as variáveis

Imprimir na tela variáveis sozinhas não é muito útil. É necessário, às vezes, integrá-las numa frase. Para imprimir junto palavras e uma variável, deve-se colocar as palavras entre aspas e colocar um ponto e vírgula de cada lado da variável. Se quisermos que a informação se destaque, basta utilizarmos vírgula no lugar de ponto e vírgula.

No decorrer de um programa é possível mudar o conteúdo de um espaço de memória. Para o computador, no exemplo apresentado no desenho acima, significa somar 1 no número rotulado X e adicionar "DA" às letras rotuladas C$

Cada vez que você mandar o computador imprimir as variáveis, ele apresentará as novas palavras e números guardados nos espaços de memória.

Você pode, da mesma forma, efetuar operações com as variáveis. O computador vai buscar os números guardados nos espaços de memória e executar o cálculo.

Quebra-cabeças

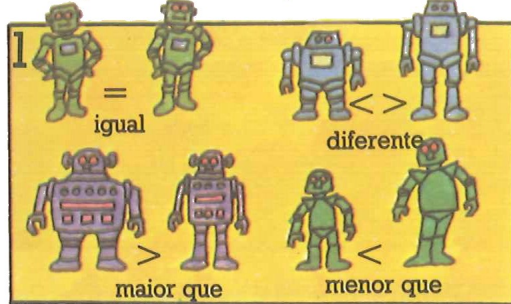
1. Faça um programa para adicionar números às variáveis do programa à esquerda, de forma que as respostas sejam 100 e 1 e estejam apresentadas na mesma linha, separadas por um espaço.

2. Modifique as linhas 30 e 40, de forma que elas apresentem os números, o que você está fazendo com eles, e os resultados: por exemplo, "7 vezes 9 são 63".

3. Mude a sua resposta ao quebra-cabeça da página 15, para que ele imprima o seu nome e a mensagem em uma linha.

Como os computadores fazem comparações

Uma das funções mais úteis do computador é comparar informações, e daí tirar conclusões. Para isto se utiliza da instrução IF... THEN (se... então)

```
2
IF A=B THEN PRINT "ELES SAO IGUAIS"
IF A>B THEN PRINT "A E MAIOR"
IF A<B THEN PRINT "A E MENOR"
IF A<> B THEN PRINT "ELES NAO SAO
                          IGUAIS"
```

O computador pode fazer diferentes testes para comparar informações. Os símbolos utilizados são mostrados acima. Ele pode determinar se dois elementos são idênticos ou diferentes, se um é maior ou menor que outro. As linhas acima mostram como combinar os símbolos com a instrução IF... THEN, para efetuar comparações entre elementos. É possível comparar qualquer tipo de elementos: palavras, números ou variáveis.

3 Programa meteorológico

```
10 PRINT "COMO ESTA O TEMPO HOJE"
20 INPUT W$
30 IF W$="CHUVA" THEN PRINT
      "LEVE GUARDA-CHUVA"
40 IF W$="SOL" THEN PRINT "QUE BOM"
50 END
```



```
4
RUN
COMO ESTA O
TEMPO HOJE
? SOL
QUE BOM
RUN
COMO ESTA O
TEMPO HOJE
? CHUVA
LEVE GUARDA-CHUVA
```

Apresentamos acima um programa usando IF... THEN. Na linha 20 o computador guarda a palavra que você entrou na variável W$. Então, na linha 30 e 40, ele verifica se a palavra em W$ é a mesma de "SOL"ou "CHUVA". Se for, ele imprimirá uma das respostas. Se você tentar entrar uma palavra diferente, nada acontecerá; é necessário modificar as palavras da linha 30 e 40.

5 Programa de idade

```
10 PRINT "QUANTOS ANOS VOCE
   TEM"
20 INPUT A
30 IF A>16 THEN PRINT "VELHO"
40 IF A<16 THEN PRINT "JOVEM"
50 IF A=16 THEN PRINT "PERFEITO"
RUN
QUANTOS ANOS VOCE TEM
? 16
PERFEITO
```

6 Aula de inglês

```
10 PRINT "COMO SE DIZ VERMELHO EM
   INGLES"
20 INPUT A$
30 IF A$="RED" THEN PRINT "CERTO"
40 IF A$<> "RED" THEN PRINT "ERRADO, A
   RESPOSTA E RED"
RUN
COMO SE DIZ VERMELHO EM INGLES
? BLUE
ERRADO, A RESPOSTA E RED
```

Neste programa de idade, o computador compara A com o número 16. Se for menor que 16, ele imprime "jovem"; maior, imprime "velho"; se for igual a 16, então imprime "perfeito". No programa à direita, ele imprimirá "CERTO" se você responder "RED", se não ele dirá "ERRADO , A RESPOSTA É RED".

Quebra-cabeça. Faça um programa tabuada em que o computador apresenta uma conta, e dependendo da sua resposta imprime "CERTO" ou dá a resposta correta.

Desvios nos programas

```
1
IF A=6 THEN LET A$= "SEIS"
IF X=Y-2 THEN LET Z = 0
IF S=T THEN STOP
IF R<10 THEN GOTO 30
```


Estas duas linhas desviam para outras partes do programa.

```
2
10 INPUT K$
20 IF K$= "SIM" THEN GOTO 100
30 IF K$= "NAO" THEN GOTO 200
100 PRINT "VOCE DISSE SIM"
110 STOP
200 PRINT "VOCE DISSE NAO"
210 END
```

Pode-se dar praticamente qualquer ordem ao computador depois da instrução THEN. Uma instrução útil é fazer o computador ir para uma outra linha do programa (na maioria dos computadores, é possível dispensar a instrução GOTO). É necessário introduzir a instrução STOP em programas com GOTO, para se evitar que ele repita infinitamente o mesmo programa.

Programa matemático

```
10 PRINT "ENTRE UM NUMERO"
20 INPUT A
30 PRINT "ENTRE OUTRO NUMERO"
40 INPUT B
50 PRINT "O QUE VOCE QUER FAZER"
60 PRINT "SOMAR, SUBTRAIR,
   MULTIPLICAR"
65 PRINT "DIVIDIR OU PARAR"
70 INPUT C$
80 IF C$ = 56 "SOMAR" THEN PRINT
   A + B
90 IF C$= "SUBTRAIR" THEN PRINT A - B
100 IF C$= "MULTIPLICAR" THEN
    PRINT A*B
110 IF C$ = "DIVIDIR" THEN
    PRINT A/ B
120 IF C$ = "PARAR" THEN
    STOP
130 GOTO 10
```

```
RUN
ENTRE UM NUMERO
? 17
ENTRE OUTRO NUMERO
? 184
O QUE VOCE QUER FAZER
SOMAR, SUBTRAIR,
MULTIPLICAR, DIVIDIR OU
PARAR
? SOMAR
201 <-------- Resposta do computador
ENTRE UM NUMERO
?
```

O PROGRAMA SOMENTE IRÁ PARAR COM A INSTRUÇÃO STOP

Neste programa os números são guardados em A e B e as ordens em C$. Nas linhas 80 até 120, o computador compara C$ com cinco palavras diferentes. Quando ele encontra a palavra certa, executa a instrução. Quando a palavra não coincide, ele salta a linha.

Programa "Qual é a sua idade"

```
1
10 PRINT "QUAL E A
   MINHA IDADE"
20 INPUT A
30 IF A < > 14 THEN PRINT
   "TENTE DE NOVO"
40 IF A < > 14 THEN
   GOTO 20
50 PRINT "CERTO"
60 END
```

```
2
RUN
QUAL E A MINHA IDADE
? 15
TENTE DE NOVO
? 14
CERTO
```

```
3
QUAL E A MINHA IDADE
? 15
MAIS JOVEM QUE ISTO
? 13
MAIS VELHO QUE ISTO
? 14
CERTO
```

PROGRAME PARA DAR ESTE RESULTADO.

Este programa se repetirá até que A seja igual a 14. Quando A for igual a 14, o computador saltará as linhas 30 e 40 e imprimirá na tela "CERTO". Você pode alterar este programa de maneira a mostrar os resultados do desenho da figura 3?

Programas em BASIC

Os programas apresentados nestas duas páginas usam a maior parte do que aprendemos até agora. O primeiro programa é um jogo espacial para duas pessoas jogarem num computador. Se você não tem um computador, estude os programas e tente entender o seu funcionamento.

Comando espacial

```
10  PRINT "POSICAO DO INIMIGO:
    LATITUDE"
20  INPUT A
30  PRINT "POSICAO DO INIMIGO:
    LONGITUDE"
40  INPUT B
50  CLS
60  PRINT "POSICAO DO COMANDO: LATITUDE"
70  INPUT C
80  PRINT "POSICAO DO COMANDO: LONGITUDE"
90  INPUT D
100 CLS
110 LET X = SQR ((A-C) * (A-C) +
    (B-D) * (B-D))
120 PRINT "VOCES ESTAO AGORA A"
130 PRINT X; "CASAS UM DO OUTRO"
140 IF X < 1.5 THEN PRINT "INIMIGO
    LOCALIZADO"
150 IF X < 1.5 THEN STOP
155 PRINT "DE SUAS NOVAS POSICOES"
160 GOTO 10
170 END
```

Nas linhas 10 a 40, o computador guarda as coordenadas do inimigo em A e B.

A linha 50 faz desaparecer as coordenadas do inimigo.

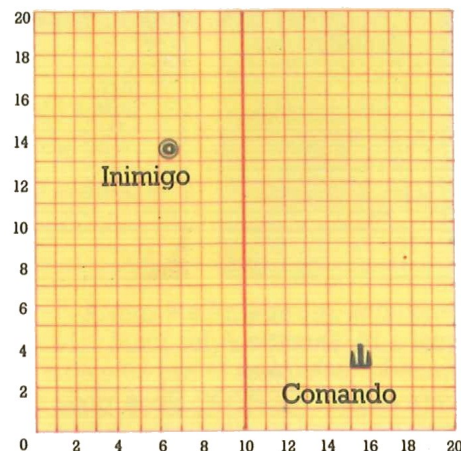
O computador guarda as coordenadas do comando em C e D.

Esta linha calcula a distância que separa os combatentes e memoriza a resposta em X.

Se X for menor que 1.5 o jogo pára. Se for maior, recomeça.

Neste jogo, um comando tentará localizar o inimigo. Cada jogador desenha um mapa secreto no qual marca a sua posição (como você pode ver abaixo). Ao informar ao computador as coordenadas, ele determina a distância que separa os adversários. Os cálculos do computador ajudam os jogadores a escolher a próxima jogada.

Regras do jogo

O mapa secreto: cada jogador desenha uma grade de 20 x 20 quadrados e os numera como mostramos à direita. Os inimigos ficam à esquerda da grade e o comando à direita. A cada jogada, cada um pode-se mover 2 casas, não importando a direção. Aí se indica ao computador as novas posições. Quando a distância que separa as espaçonaves for inferior a 1.5 quadrado, o comando localizou o inimigo e venceu o jogo.

Como fazer o computador parecer inteligente

Neste programa, o computador parece manter um diálogo. Você pode ver como o programa funciona nas figuras ao pé da página. Utiliza-se aqui a função INPUT de maneira ligeiramente diferente, o que faz o programa ficar menor e mais fácil de ser lido.

2
No CP200 ou TK 82/83/85 você tem que bater 10 PRINT "DE UM NUMERO" 15 INPUT N
RUN
DE UM NUMERO ? 10
E UM OUTRO ? 8
10 VEZES 8 IGUAL A 80

Na maioria dos computadores (com exceção do ZX 81, e os seus similares nacionais, TK 82/83/85 e CP 200) você pode fazer a linha INPUT ficar mais clara, colocando uma frase entre aspas antes da variável. Quando o programa é acionado, a frase aparece no vídeo seguida de um ponto de interrogação.

O programa

```
5   LET C = 0
10  PRINT "EU GOSTARIA DE CONVERSAR COM VOCE"
20  INPUT "DIGA ALGUMA COISA ENGRAÇADA QUE
    ACONTECEU COM VOCE ESTA SEMANA"; A$
30  READ B$
40  PRINT B$;  <-- o ponto e vírgula impede a troca de linha
50  INPUT C$   <-- sua resposta fica guardada em C$
60  LET C = C + 1
70  IF C = 6 THEN GOTO 100
80  GOTO 30
90  DATA POR QUE, POR QUE ISSO
95  DATA PODE EXPLICAR, POR QUE
98  DATA DIGA POR QUE, POR QUE RAZAO
100 PRINT "ASSIM, A VERDADEIRA RAZAO POR QUE"
110 PRINT "      "; A$
120 PRINT "FOI PORQUE"
130 PRINT "      "; C$
140 PRINT "QUE BARATO!"
150 PRINT "ACIONE-ME DE NOVO PARA
    CONTINUARMOS NOS DIVERTINDO"
160 END
```

Esta é a nova maneira de usar INPUT. Sua resposta é guardada em A$.

A linha 30 faz o computador procurar a instrução DATA mais próxima e guardá-la em B$.

A variável C nas linhas 60 e 70 é um contador. Ela anota quantas vezes o programa foi repetido. Quando C = 6 todos os dados foram usados e o computador passa à linha 100.

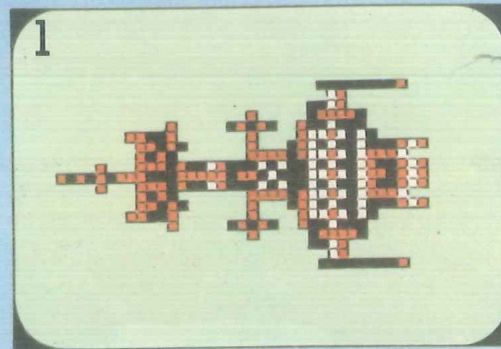
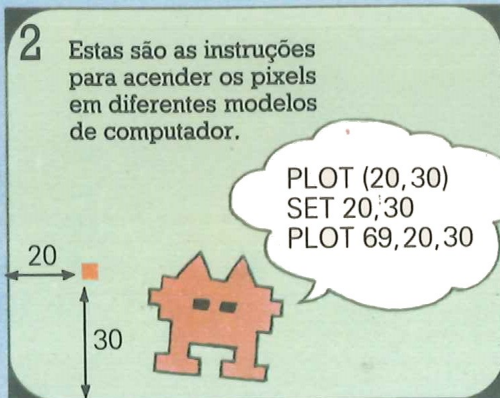
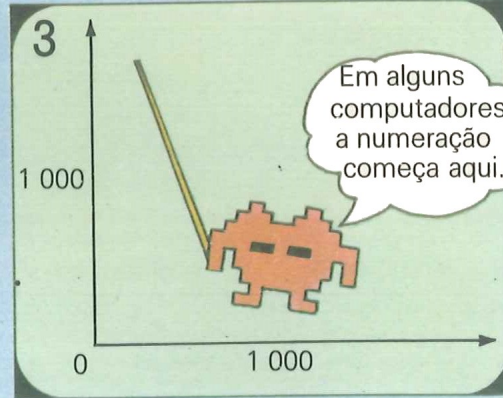
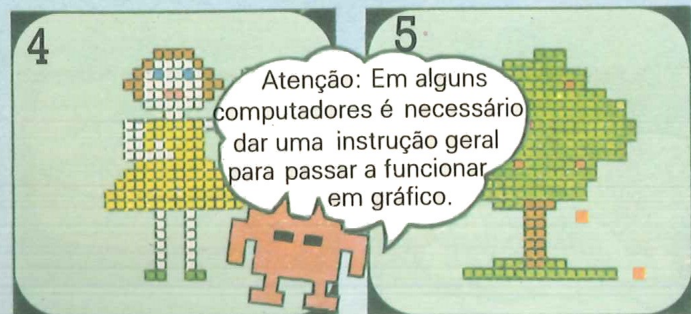
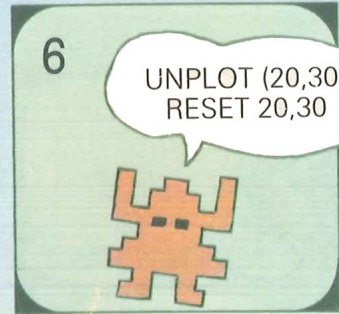
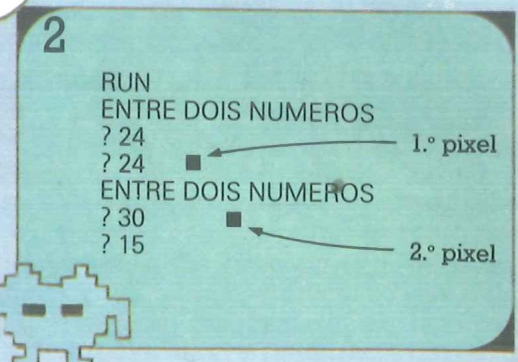
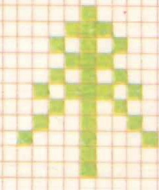
A linha 80 faz o computador voltar a 30 e troca o dado de B$ pelo próximo item na lista de DATA.

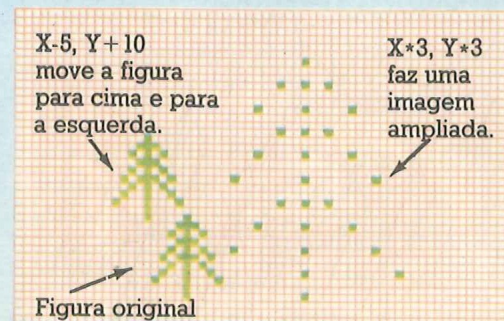
Estes espaços nas linhas 110 e 130 deixam a resposta destacada no vídeo. O número de espaços deixado não tem importância.

Como funciona

ASSIM, A VERDADEIRA RAZAO POR QUE
EU CAI NUM BURACO
FOI PORQUE
EU NAO QUERIA QUE MEU AMIGO VISSE O MEU SORVETE
QUE BARATO!
ACIONE-ME DE NOVO PARA CONTINUARMOS NOS DIVERTINDO

Desenhando

Um computador faz figuras acendendo pequenos retângulos na tela. Cada retângulo, ou pixel, responde a uma instrução separada. Nestas duas páginas, vamos aprender a usar o BASIC para desenhar na tela. Alguns computadores permitem fazer desenhos a cores. Verifique o seu manual de instruções.

Você pode normalmente ver os pixels na tela do computador. Um computador com grande memória, entretanto, pode fazer figuras com milhares de pequenos pontos. Estas figuras são chamadas de gráficos de alta resolução.

Estas instruções permitem que se acendam os pontos em diferentes computadores. A instrução que permite acender um pixel varia de aparelho para aparelho, mas ela se apresenta geralmente sob a forma de PLOT (X, Y). X e Y são as coordenadas do pixel.

Um computador de alta resolução gráfica oferece mais de 1 000 x 1 000 pontos na tela. Um computador menos potente tem aproximadamente 60 x 40 pontos. (Verifique a dimensão da sua tela, já que você arrisca a plotar fora dos limites.)

Os desenhos feitos por computador são chamados de grafismos. Em certos modelos você necessita dar uma instrução especial para funcionar em gráfico. Para apagar um pixel, utilize a instrução UNPLOT (X, Y). Verifique no seu manual de instruções.

Um programa para plotar

Este pequeno programa pede dois números e depois plota os pixels cujas coordenadas são estes dois números. Verifique se os números que você entrou estão dentro dos limites do seu computador. A linha 50 faz o programa se repetir infinitamente; a única maneira de interrompê-lo é apertando a tecla BREAK (ou aquela correspondente no seu computador). Será que você pode introduzir um contador (veja a pág. 21) para fazê-lo funcionar seis vezes?

Antes de mais nada, é necessário traçar a figura num papel quadriculado e determinar as coordenadas de cada ponto.

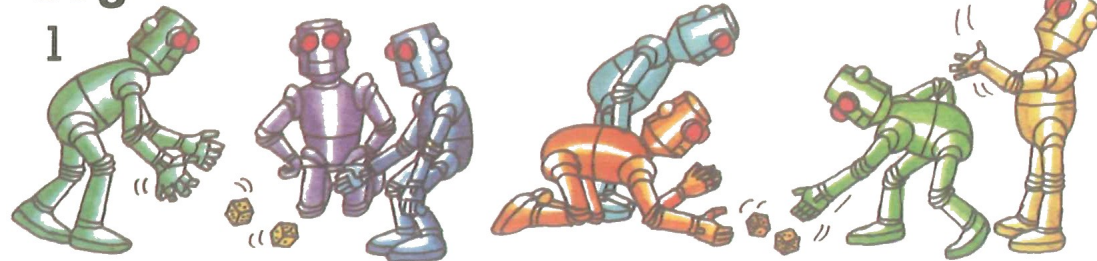
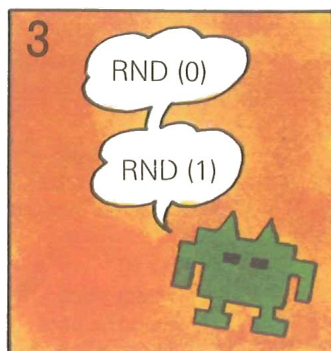
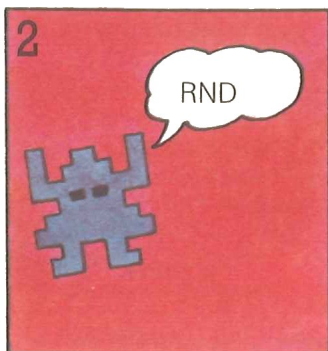
Depois, você pode fazer um programa para plotar todos os pontos. A partir dos valores de origem de X e Y, e os aumentando ou os diminuindo, e repetindo parte do programa, pode-se fazer o computador desenhar a seqüência de pontos como mostrado acima.

Depois de escrever o programa, é fácil mover a figura, alterando os números. Você pode movê-la para um lugar diferente na tela, modificando os pontos iniciais, ou multiplicar todos os números por 3 para obter uma imagem ampliada.

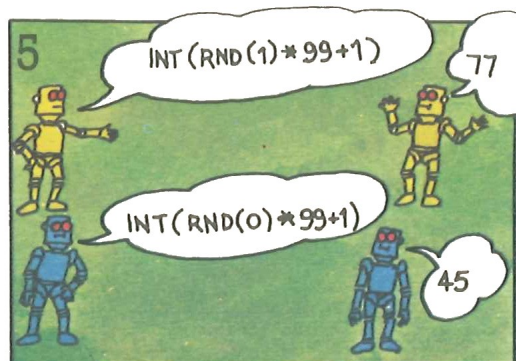
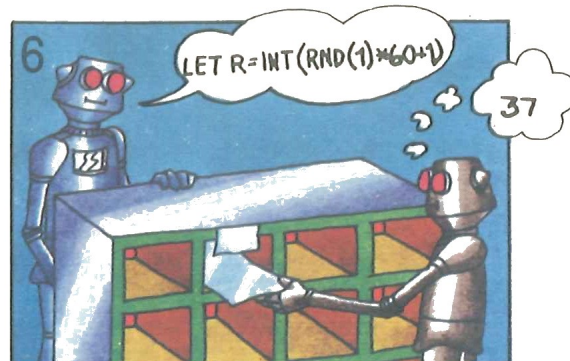
Uma outra maneira

Na realidade, você só pode fazer figuras bastante simples com PLOT. Para desenhar algo mais complicado, é necessário um equipamento especial. Colocando-se o desenho na superfície do equipamento e com a ajuda de um sensor eletrônico, redesenha-se por cima. Automaticamente as coordenadas são fornecidas ao computador.

Quebra-cabeça. Faça um programa para desenhar as suas iniciais na tela. Você encontrará um exemplo na página 44.

Jogos

1

Quando se lança um dado, não se sabe jamais qual o número que sairá. As chances são as mesmas para qualquer número de um a seis. Da mesma forma, o computador pode gerar números imprevisíveis, ou seja, aleatórios. A função RND produz números aleatórios. Às vezes, por acaso, um mesmo número se repete, na mesma série de números, mas a probabilidade de sair cada número é igual.

Para fazer o computador gerar um número aleatório, usa-se a instrução RND. Em alguns modelos, é necessário colocar o algarismo 1 ou 0 entre parênteses. Verifique o seu manual.

A instrução RND sempre gera um número menor que um. Em alguns computadores, você pode colocar um número entre parênteses depois de RND. Obtém então um número inteiro entre 1 e o número entre parênteses.

Em outros computadores utiliza-se a instrução INT (abreviação de inteiro, significando número inteiro), seguido da instrução RND (ou RND (0) ou RND (1), dependendo do modelo). Em seguida, multiplica-se pelo número maior que você deseja obter, e soma-se um para que o resultado seja superior a zero.

Esta instrução significa escolher um número aleatório e guardá-lo na variável R. Os programas deste livro usam INT (RND (1) * 60 + 1) para significar um número aleatório entre 1 e 60. Pode ser necessário converter esta instrução para o seu tipo de computador.

Quebra-cabeça. Faça um programa para o computador escolher um número aleatório entre 10 e 20.

Ataque espacial

Este é um programa para jogo usando números aleatórios. Neste jogo você está viajando numa nave espacial que está sendo atacada por uma esquadrilha de caças extraterrestres. O computador da nave localiza o inimigo e dá a sua posição. Para destruir cada nave é necessário determinar a distância de tiro, multiplicando as coordenadas e entrando o resultado no computador.

```
10 LET C = 0
20 LET A = INT (RND(1)*20 + 1)
30 LET B = INT (RND(1)*20 + 1)
40 PRINT "POSICAO DOS
   EXTRATERRESTRES"
45 PRINT A, B, "FOGO"
50 INPUT X
60 LET C = C+1
70 IF X = A * B THEN PRINT "NAVE
   INIMIGA DESTRUIDA"
80 IF X <> A * B THEN PRINT "ERROU"
90 IF C<6 THEN GOTO 20
100 END
```

Linha 10: C permite contar o número de repetições do programa. A cada passagem, a linha 60 soma 1 a C.

Linhas 20–30: Estas duas linhas geram os números aleatórios que determinam as coordenadas dos extraterrestres, que ficam memorizadas em A e B.

Linhas 70–80: Estas linhas conferem se a sua resposta está certa.

Linha 90: Esta linha aciona de novo o programa se C for inferior a 6.

Rodando o programa

O quadro à direita mostra o que se passa quando o programa é acionado. Se a sua resposta for correta (igual ao resultado da multiplicação das duas coordenadas), o computador imprimirá "nave inimiga destruída". Se a sua resposta estiver errada, ele imprimirá "errou".

```
RUN
POSICAO DOS
EXTRATERRESTRES
17    3 FOGO
? 41
ERROU
POSICAO DOS
EXTRATERRESTRES
11    5 FOGO
? 55
NAVE INIMIGA
DESTRUIDA
```

A vírgula da linha 45 espaça os número assim.

Quebra-cabeça

Coloque um outro contador no programa para registrar o número de tiros certos, e imprima 10 no placar final do jogo. Você precisa introduzir uma variável P, dar-lhe no início o valor zero e aumentá-la de 1 a cada acerto.

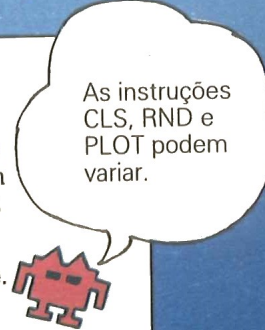
Desenho aleatório

```
5 CLS
10 LET X = INT(RND(1)*30+1)
20 LET Y = INT(RND(1)*30+1)
30 PLOT (X, Y)
40 GOTO 10
```

Linha 5: Esta instrução limpa a tela.

Linhas 10–20: Verifique se os valores dados aos números aleatórios correspondem à capacidade do seu computador.

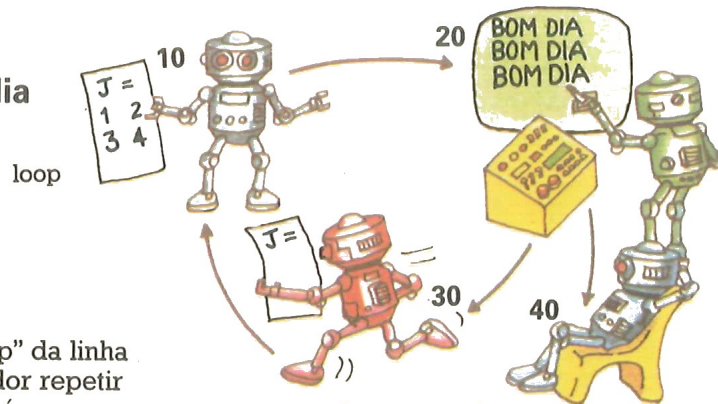
Linha 40: Esta instrução permite ao programa repetir-se infinitamente.

As instruções CLS, RND e PLOT podem variar.

Este programa usa números aleatórios para plotar pontos na tela. As linhas 10 e 20 selecionam números entre 1 e 30 que são guardados em X e Y. A linha 30 plota o ponto com as coordenadas X e Y. Para interromper o programa aperte a tecla BREAK ou ESCAPE, ou entre o comando correspondente no seu computador.

Fazendo "loops"

Em diversas ocasiões, é necessário que o computador repita várias vezes a mesma coisa. Você viu, na página 21, que se pode obter este efeito utilizando GOTO e uma variável que faz a função de um contador. Uma outra maneira é utilizando-se as instruções FOR... TO e NEXT. Isto é o que chamamos fazer um "loop".

1 O "loop" do bom dia

```
10 FOR J = 1 TO 6
20 PRINT "BOM DIA"
30 NEXT J
40 END
```

loop

O programa forma um "loop" da linha 10 à 30, que faz o computador repetir 6 vezes a linha 20. A letra J é uma variável que na linha 10 recebe o valor 1 na primeira passagem do programa, 2 na segunda e assim por diante até 6. Quando J é igual a 6, o computador passa para a linha 40.

2 Programa bobo de adição

```
10 FOR J = 1 TO 8
20 PRINT "2 MAIS 2 SÃO 5"
30 NEXT J
40 PRINT
50 PRINT "BRINCADEIRA!"
60 END
```

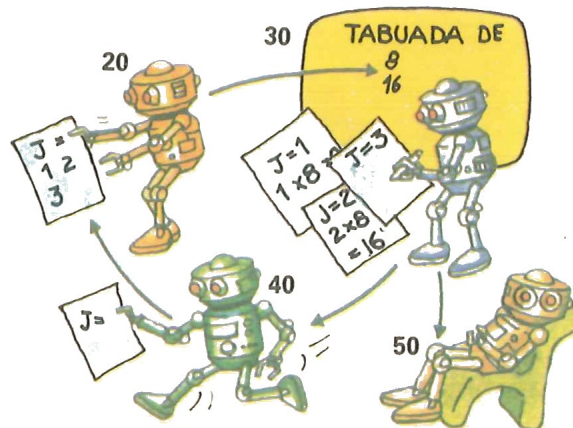
loop

Neste programa, o "loop" das linhas 10 a 30 faz repetir a linha 20 oito vezes. Cada vez que passa pela linha 20, imprime outra vez a soma errada.

Depois de repetir oito vezes, o computador prossegue com o resto do programa. A linha 40 serve apenas para pular uma linha.

3 Programa tabuada de 8

```
10 PRINT "TABUADA DE 8"
20 FOR J = 1 TO 10
30 PRINT J * 8
40 NEXT J
50 END
```

loop

Neste exemplo, J serve para contar o número de repetições e também como elemento da operação J * 8. Na linha 20, o computador dá o valor 1 para J, depois 2, 3, assim por diante, até 10. A linha 30 aproveita o valor dado a J, multiplica-o por 8 e imprime o resultado. Finalmente, a linha 40 remete o computador à linha 20, onde J recebe um novo valor.

Para desenhar

FOR... NEXT é útil, também, para criar desenhos simples e repetitivos. O programa que corresponde à figura acima é muito longo para reproduzir aqui, mas seria semelhante ao seguinte:

```
10 FOR I = 1 TO 45
20 Desenhe um retângulo e mude
   um pouco as suas coordenadas
   a cada vez.
30 NEXT I
40 END
```

Uso de STEP

Às vezes, é interessante que o valor de J seja alterado, não de 1 em 1, mas de 3 em 3 ou 7 em 7. Para este fim, usa-se a instrução STEP. No programa abaixo, STEP-1 faz decrescer de 1 o valor de J a cada passagem do programa pelas linhas 10 a 40.

O computador guloso

O número 2 interrompe o "loop" depois de J = 2

```
  5 CLS
 10 FOR J = 7 TO 2 STEP - 1
 20 PRINT "SOBRARAM"; J; "DOCES"
 30 NEXT J
 40 PRINT
 50 PRINT "EU VOU EXPLODIR"
 60 FOR K = 1 TO 1000
 70 REM: NÃO FAZ NADA
 80 NEXT K
 90 PRINT
100 PRINT "BUUUUM"
```

"loop"

"loop"

Existem dois "loops" neste programa. O das linhas 10 a 30 faz imprimir 6 vezes a linha 20, cada uma com o valor de J reduzido de 1. O "loop" das linhas 60 a 80 não faz nada. Apenas, ao fazer K assumir os valores de 1 a 1000, provoca uma pausa no desenvolvimento do programa. As linhas começando por REM não são levadas em conta pelo computador. São úteis para lembrar a você o que o programa está fazendo.

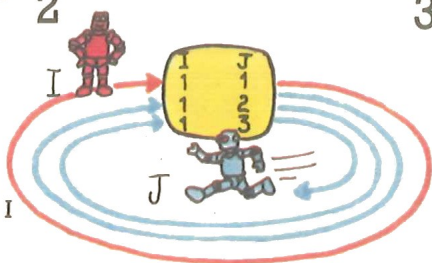
Quebra-cabeça

1. Transforme o programa da tabuada de 8 para que imprima "1x8 = " e o resultado.

2. Faça um programa para a tabuada de "N", que você possa utilizar para qualquer número informado ao computador. Inicialmente, o computador deverá pedir o valor de N. Depois, através de um "loop", ele calcula e imprime a tabuada de multiplicação.

Alguns truques com "loops"

Apresentamos, aqui, outros programas com "loops". Abaixo, você verá como utilizar um "loop" dentro de outro "loop" para repetir várias coisas ao mesmo tempo. São os chamados "loops encaixados".

```
5 PRINT "I", "J"
10 FOR I = 1 TO 3
20 FOR J = 1 TO 3
30 PRINT I, J
40 NEXT J
50 NEXT I
60 END
```

I	J
1	1
1	2
1	3
2	1
2	2
2	3
3	1
3	2
3	3

Este programa tem um "loop" do I e um "loop" do J. O "loop" do J está encaixado no "loop" do I. A cada passagem do "loop" do I, o "loop" do J se repete três vezes, imprimindo a cada volta um novo valor para J. No quadro acima, são mostrados os resultados deste programa.

O relógio do computador

```
5 CLS
10 LET M = 0
20 LET S = 0
30 FOR M = 0 TO 59
40 FOR S = 0 TO 59
50 PRINT M; ":"; S
60 CLS
70 NEXT S
80 NEXT M
90 END
```

Grilo no "loop"

```
10 FOR I = 1 TO 4
20 FOR J = 1 TO 4
30 PRINT I
40 PRINT J
50 NEXT I
60 NEXT J
```

As duas partes de um "loop encaixado" devem estar dentro do outro "loop"

Dentro do computador encontra-se um relógio eletrônico que determina o seu ritmo de funcionamento. O relógio emite entre 1 a 4 milhões de pulsos por segundo. Este programa faz o computador agir como um relógio digital. Usa-se dois "loops", um para contar os segundos, o outro para os minutos. O "loop" interno é executado 59 vezes para cada passagem no "loop" dos minutos. Coloque um "loop" de espera e vá trocando o seu número até fazer o computador contar os segundos corretamente.

Teste da função RND

```
10 FOR I = 1 TO 1.000
20 LET R = INT (RND (1) * 6 + 1)
30 IF R = 1 THEN LET A = A + 1
40 IF R = 2 THEN LET B = B + 1
50 IF R = 3 THEN LET C = C + 1
60 IF R = 4 THEN LET D = D + 1
70 IF R = 5 THEN LET E = E + 1
80 IF R = 6 THEN LET F = F + 1
90 NEXT I
100 PRINT "TERMINOU"
110 PRINT A, B, C
120 PRINT D, E, F
130 END
```

O programa verifica o funcionamento da função RND. O "loop" das linhas 10 a 90 faz o computador obter um número entre 1 e 6, mil vezes. O computador guarda nas variáveis A, B, C, D, E e F o número de vezes que cada número saiu e depois imprime os resultados.*

* Em certos computadores, como o SINCLAIR ZX81 os TK 81/82 e 83, é necessário colocar algumas linhas no começo do programa para inicializar as variáveis.

Repetição de desenhos

Este programa usa os "loops encaixados" para repetir um mesmo desenho através do vídeo. Ele parece um pouco complicado, mas se você o ler atentamente, descobrirá rapidamente como funciona. O formato do desenho é definido pelos números aleatórios, e será diferente a cada vez que você rodar o programa.

Para computadores com alta resolução gráfica, use números aleatórios maiores.

```
5 CLS
10 LET A = INT (RND (1) *6+1)
20 LET B = INT (RND (1) *7+1)
30 LET C = INT (RND (1) *6+1)
40 LET D = INT (RND (1) *4+1)
50 INPUT "QUAL E A DEFINICAO
   DO SEU VIDEO: LARGURA"; L
60 INPUT "ALTURA"; H
65 CLS
70 FOR I = 0 TO H STEP H/6
80 FOR J = 0 TO L STEP L/6
90 PLOT (J+A, I+B)
100 PLOT (J+A, I+C)
110 PLOT (J+C, I+D)
120 PLOT (J+B, I+D)
130 NEXT J
140 NEXT I
150 END
```

loop do J

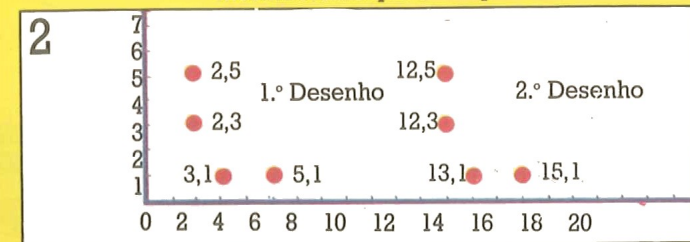
loop do I

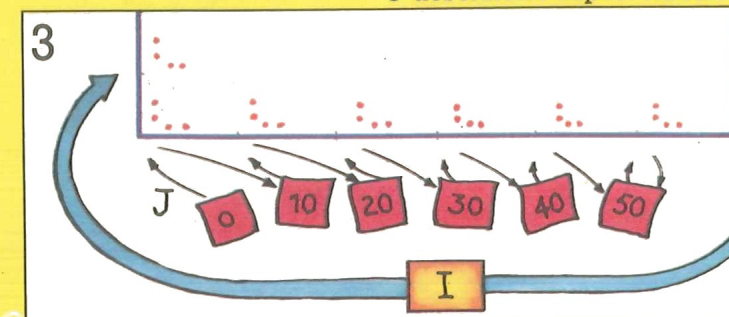
Estas linhas escolhem os números aleatórios, que são guardados nas variáveis A, B, C e D.

As linhas 50 e 60 perguntam a largura (L) e a altura (H) da sua tela.

O "loop" do I conta o número de vezes que o desenho é repetido na tela. A cada vez, I é aumentado pela altura da tela (H), dividida por 6.

A cada vez que os "loops" são repetidos, as linhas 90 e 120 fazem o computador plotar 4 pontos, usando os valores de I e J somados de números aleatórios.

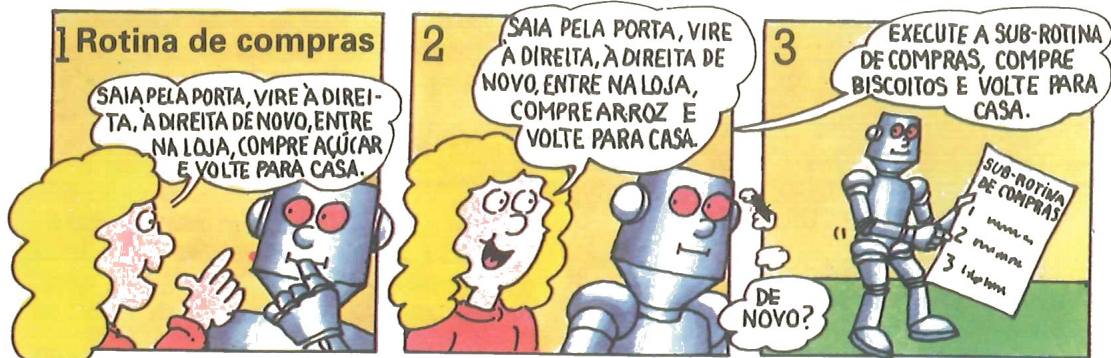
O "loop" do J conta o número de vezes que o desenho é repetido na tela. Funciona da mesma forma que o "loop" do I.

Imagine que o computador escolheu os números aleatórios 2, 5, 3 e 1 e que tanto a largura como a altura da tela são 60.

Na primeira execução do programa, I e J são iguais a 0, e assim o computador plota o primeiro desenho usando apenas os números aleatórios. A linha 130 faz com que ele volte para encontrar o valor seguinte de J que é 0 + 60/6, ou seja, 10. Assim, ele plota o segundo desenho usando os números aleatórios somados a 10. O desenho se repete na tela.

Se este programa não funcionar, tente trocar os números de L e H por números menores.

O computador repete o "loop" de J seis vezes, cada vez somando 10 ao valor anterior de J, e assim vai movendo o desenho ao longo da tela. Volta então para buscar um novo valor para I, ou seja, 10. J volta a ser 0 de novo e o computador plota uma nova linha de desenhos, com I = 10 e J aumentando de 10 em 10.

Quebra-cabeça. Será que você pode fazer um programa onde a figura que se repete na tela é a de um extraterrestre? Na página 45, você encontrará algumas dicas para ajudá-lo.

Sub-rotinas

Uma sub-rotina é uma espécie de um miniprograma dentro de um programa. Ela executa uma tarefa específica, como somar ou manter um placar, e você pode mandar o computador executá-la sempre que for necessário. Isto economiza linhas de programação e faz o programa menor, mais fácil de ler e entrar no computador.

Suponha que você tenha um robô que você possa programar para ajudá-lo em casa. Se você quiser alguma coisa do mercado, você tem que lhe dar instruções precisas de como agir. Cada vez que você precisar que o robô compre alguma coisa, você tem que lhe dar as mesmas instruções. Não seria muito mais simples dar ao robô uma sub-rotina de compras e, quando necessário, fazer com que ele a execute?

4 Programa de compras

```
10  PRINT "O QUE VOCE QUER DO MERCADO"
20  INPUT X$
30  GOSUB 100
40  PRINT "ALGUMA COISA MAIS"
50  INPUT M$
60  IF M$ = "SIM" THEN GOTO 10
70  STOP
100 REM: SUB-ROTINA DE
    COMPRAS
110 PRINT "SAIA, VIRE A DIREITA"
120 PRINT "A DIREITA DE NOVO, ENTRE NA LOJA"
130 PRINT "COMPRE"; X$; "VOLTE PARA CASA"
140 RETURN
```

A linha 30 envia o computador à primeira linha da sub-rotina.

Você precisa da palavra STOP para evitar que o computador vá executar a sub-rotina.

É útil colocar um REM para lembrar o que a sub-rotina faz.

Esta instrução remete o computador para a linha 40 — a linha depois do GOSUB.

Se você esquecer o RETURN, você arranja um GRILO.

Em BASIC, para mandar o computador executar uma sub-rotina, você usa a instrução GOSUB, e a instrução RETURN no fim da sub-rotina. GOSUB deve ser seguida do número da primeira linha da sub-rotina. RETURN não precisa de número de linha. O computador, automaticamente, retorna para a instrução imediatamente após o GOSUB. Você pode enviar o computador para uma sub-rotina quantas vezes quiser.

Programas com GOSUB

Uma sub-rotina é muito útil para executar uma tarefa que se repete diversas vezes, em diferentes estágios do programa. Apresentamos abaixo alguns programas com sub-rotina.

Programa de divisão

```
50  INPUT A
60  INPUT B
70  GOSUB 250
80  PRINT "A DIVIDIDO
    POR B ="; A/B
90  GOTO 50
250 REM: SUB-ROTINA PARA PARAR
260 IF A = 0 AND B = 0 THEN STOP
270 RETURN
```

Esta sub-rotina permite sair do programa principal. Se você quiser parar de dividir, entre com 0 nas linhas 50 e 60. Este programa dispensa STOP antes da sub-rotina, já que a linha 90 o remete à linha 50.

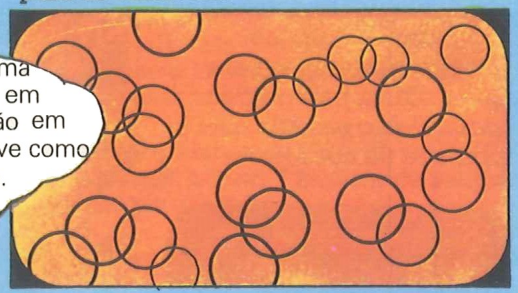
Programa de conversão

```
100 INPUT "DISTANCIA"; K
110 INPUT "TEMPO"; T
120 GOSUB 200
130 PRINT "VELOCIDADE MEDIA E"
140 PRINT K/T; "KM/H E"; M/T; "MPH"
150 STOP
200 REM: SUB-ROTINA PARA
         CONVERTER KM EM MILHAS
210 LET M = K/1.609
220 RETURN
```

Esta é uma sub-rotina para converter quilômetros em milhas. Você pode usá-la em diversos programas, mas tenha o cuidado de usar o mesmo nome para as variáveis.

Programa de Círculos

1. Centro do Círculo = X, Y
2. Raio do Círculo = R
3. Cor = X
4. Gosub 10
5. Goto 1
10. Rem: Sub-rotina para desenhar o círculo
11. Desenho um círculo com centro X, Y; raio R e cor X
12. Return

Este programa está escrito em português, não em BASIC, e serve como sugestão.

Sub-rotinas são úteis em programas gráficos como este, para desenhar diagramas a partir de números obtidos na parte principal do programa. Neste programa, você poderia traçar muitos círculos diferentes mudando os números das linhas 1 a 5.

Programa de perguntas e respostas

```
5   LET C = 0
10  PRINT "QUANDO ESTAS COISAS FORAM INVENTADAS?"
20  READ C$, F
30  PRINT C$
40  INPUT A
50  LET C = C + 1
60  IF C = 3 THEN STOP
70  GOSUB 100
80  GOTO 10
100 REM: SUB-ROTINA DAS RESPOSTAS
110 IF ABS (A—F)<10 THEN PRINT "CERTO"
120 IF ABS (A—F)> 10 THEN PRINT "ERRADO"
130 PRINT "TENTE OUTRA"
140 RETURN
200 DATA TELEFONE, 1876, IMPRENSA, 1450, BICICLETA, 1791
```

Na linha 20, o computador procura a instrução DATA e põe os primeiros itens em C$ e F.

Linha 30: Imprime a palavra memorizada em C$

O contador C pára o programa, depois de repetir três vezes, já que há apenas três dados para C$ e F.

Esta é a sub-rotina.

Cada vez que o programa é repetido, as palavras e números de C$ e F são trocados pelo próximo par de dados.

Este programa usa uma sub-rotina para conferir as respostas. As respostas corretas ficam guardadas em F e as respostas do jogador em A. Nas linhas 110 e 120 da sub-rotina, o computador compara A com F. ABS significa valor absoluto: o aparelho compara A com F sem levar em conta o sinal negativo. Se a diferença A-F for inferior a 10, ele imprime "CERTO". Se for maior que 10, imprime "ERRADO".

Brincando com palavras

A maioria dos computadores pode analisar as palavras guardadas nas variáveis. Eles podem verificar o conteúdo das variáveis, para ver se contém uma determinada palavra ou letra. Isto é útil para conferir as palavras entradas por alguém usando o programa. O computador é capaz, também, de reorganizar as letras ou palavras em uma ordem diferente e juntá-las com outras variáveis. Abaixo você verá como proceder em BASIC.

1

```
10 A$ = "EU SOU UM BURRO"
20 B$ = "SO UM LOUCO PENSA"
30 C$ = B$ + "QUE" + A$
RUN
SO UM LOUCO PENSA QUE EU SOU
  UM BURRO
```

Você pode juntar o conteúdo de duas variáveis. Você deve deixar um espaço entre aspas para que a última palavra de uma variável não se junte com a primeira da outra.

2

```
PRINT LEFT $ (B$,2)
SO
PRINT LEFT $ (B$,2) + " " + A$
SO EU SOU UM BURRO
```

Você pode também juntar partes de variáveis. LEFT (B$,2) significa tomar as duas primeiras letras à esquerda de B$.

3

```
PRINT RIGHT$ (A$,5)
BURRO
```

Para dizer ao computador para tomar as letras a partir da direita, use RIGHT$ com o nome da variável e o número de letras que você quer.

4

```
PRINT MID$ (B$,7,5)
LOUCO
```

Neste exemplo, mostramos como tomar letras no meio da frase. O primeiro número indica onde começar, e o segundo, quantas letras tomar.

Quando contar as letras, conte os espaços e a pontuação também.

5

```
10 K$ = "DING DONG!"
20 PRINT LEN (K$)
RUN
10
```

Você pode descobrir o tamanho de uma cadeia (string) — ou seja, o número de letras, espaços e símbolos nela contidos. Use para isto a função LEN.

Se A$ = "LIVRO DE COMPUTADOR" que é LEFT (A$,8)? e RIGHT$ (A$,10)? MID$ (A$,7,7)?

Para quem tem um TK 82/83/85 ou CP200

```
PRINT A$ (11 TO 15)
BURRO
PRINT B$ (13 TO 17)
PENSA
```

Isto significa tomar as letras de 11 a 15 em A$.

Os computadores da linha TK ou CP200 não têm as funções LEFT$, RIGHT$ ou MD$, mas você pode fazê-los imprimir na tela certas letras.

Código secreto

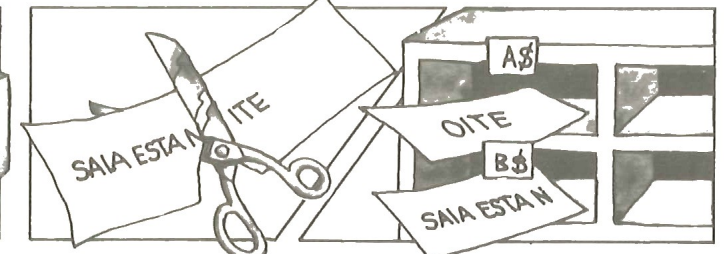
Este programa serve para codificar mensagens. Programas semelhantes, mas muito mais complexos, são usados pelos serviços secretos para codificar e decodificar mensagens.

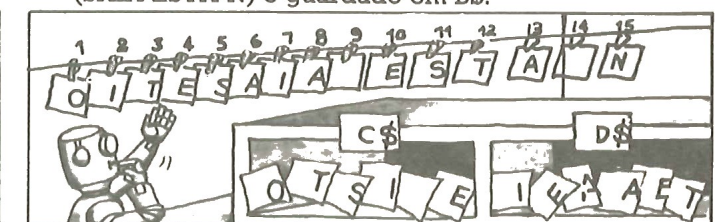
A maneira mais fácil de entender este programa é escrever num papel uma mensagem secreta, acompanhar, linha a linha, as tarefas executadas pelo computador e ir anotando os resultados passo a passo.

```
  5 LET C$ = ""
  7 LET D$ = ""
 10 PRINT "ENTRE UMA MENSAGEM
    CURTA"
 20 INPUT M$
 30 PRINT "AGORA, ENTRE UM NUMERO
    SECRETO ENTRE 2 E", LEN (M$)-1
 40 INPUT N
 50 LET A$ = RIGHT$ (M$,N)
 60 LET B$ = LEFT $ (M$, LEN (M$)-N)
 70 LET M$ = A$ + B$
 80 FOR I = 1 TO LEN (M$) STEP 2
 90 LET C$ = CS + MID$ (M$ I,1)
100 NEXT I
110 FOR J = 2 TO LEN (M$) STEP 2
120 LET D$ = D$ + MID$ (M$,J,1)
130 NEXT J
140 LET M$ = C$ + D$
150 PRINT "A MENSAGEM CODIFICADA E"
160 PRINT M$
170 END
```

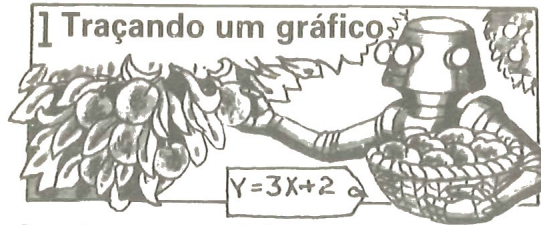
- Linhas 5 e 7: Cria duas variáveis vazias.
- Linha 30: Isto corresponde ao tamanho de sua mensagem menos 1.
- N (o número secreto) determina o número de letras a partir da direita.
- Linha 60: O tamanho de M$ menos N, a partir da esquerda corresponde ao resto das letras da mensagem M$.
- Linha 70: Troca o conteúdo de M$ para A$ + B$.
- Linha 80: De 1 ao número de letras de sua mensagem, o "loop" da linha 90 é repetido de 2 em 2, tirando uma letra da posição I de M$ e colocando-a em C$.
- Linha 140: De 2 até o número de letras de sua mensagem, o "loop" J funciona da mesma forma que o "loop" I.

Como funciona

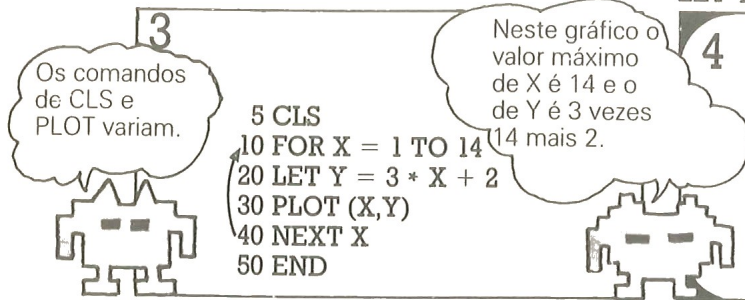
Suponha que a mensagem secreta seja "saia esta noite", e o número secreto seja 4. Estas informações estão guardadas em M$ e N.

Nas linhas 50 e 60, o computador usa o seu número secreto para dividir a mensagem. Na linha 50, ele retira 4 letras da direita da mensagem (OITE), e guarda em A$. Na linha 60, o resto da mensagem (SAIA ESTA N) é guardado em B$.

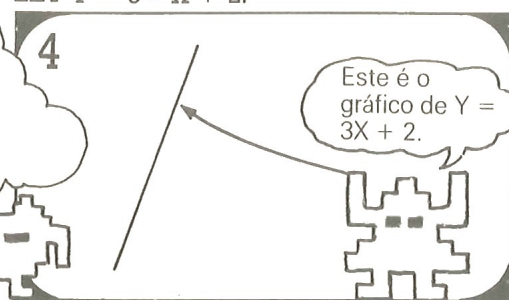
Na linha 70, ele junta A$ e B$, ou seja, inverte a mensagem.

Cada vez que o "loop" do I é repetido, uma letra de ordem ímpar é guardada em C$ (por exemplo O,T,S,I, etc.). Cada vez que o "loop" do J é repetido, as letras de ordem par são guardadas em D$ (por exemplo I,E,A,A, etc.). Ao reunir C$ com D$, a mensagem está codificada.

Gráficos e símbolos

Você pode programar o computador para apresentar a informação de diversas maneiras: palavras, números, desenhos ou gráficos. Informações complicadas podem ficar muito mais fáceis de entender, se ilustradas por gráficos, desenhos ou símbolos.

Imagine uma laranjeira cuja produção de frutas aumenta a cada ano. Isto pode ser expresso como uma equação, como Y = 3X + 2 (Y é a produção e X é a idade). Se traçarmos um gráfico, esta equação fica mais fácil de entender.

Com um computador não é difícil traçar um gráfico da maneira como Y varia em relação a X. Para isto, você precisa descobrir o valor de Y para cada valor de X, o que pode ser facilmente obtido num programa, usando a instrução LET Y = 3 * X + 2.

```
5 CLS
10 FOR X = 1 TO 14
20 LET Y = 3 * X + 2
30 PLOT (X,Y)
40 NEXT X
50 END
```


Este é o programa para traçar o gráfico. O "loop" dá à variável X todos os valores de 1 a 14. Cada vez que o "loop" é repetido, a linha 20 usa o valor de X para calcular Y e a linha 30 plota X e Y na tela. Em programas deste tipo, você deve-se certificar se os valores máximos de X e Y estão contidos na tela, caso contrário o programa não funcionará.

Computadores e matemática

Em cálculos múltiplos, tais como 3 * X + 2, o computador sempre faz as multiplicações e divisões antes das somas e subtrações. Isto significa que as respostas para as seguintes operações serão iguais:

```
PRINT 4 * 6 + 8          PRINT 8 + 4 * 6
32                       32
```

Se você quiser que o computador faça o cálculo numa ordem diferente, utilize parênteses:

```
PRINT (8 + 4) * 6
72
```

Desta vez, o computador somou 8 a 4 e multiplicou o resultado por 6.

Quebra-cabeça

PENSE NUM NUMERO
MULTIPLIQUE-O POR 2, SOME 4
DIVIDA POR 2, SOME 7
MULTIPLIQUE POR 8, SUBTRAIA 12
DIVIDA POR 4, TIRE 11
E DIGA O RESULTADO.
O NUMERO QUE VOCE ESCOLHEU
E...

Será que você pode fazer um programa para resolver este jogo de números? (Para descobrir o número escolhido, subtraia 4 do resultado e depois divida por 2.)

Programa das estações

Este programa usa um outro método para apresentar informações na tela. Ele usa símbolos para comparar o número de pessoas nascidas nas diferentes estações do ano. Você pode usar um programa como este para comparar diferentes coisas, como, por exemplo, o número de vitórias de times de futebol. Antes de escrever um programa tão longo como este, vale a pena fazer um plano do programa.

Plano do programa

Objetivo: comparar o número de pessoas nascidas no inverno, primavera, verão e outono.

INVERNO — junho/julho agosto

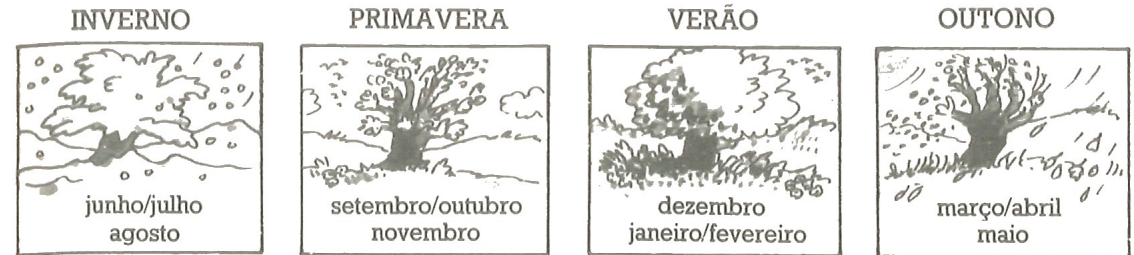
PRIMAVERA

setembro/outubro novembro

VERÃO

dezembro janeiro/fevereiro

OUTONO

março/abril maio

1. Informe ao computador as estações em que nasceram as pessoas selecionadas para a pesquisa.
2. Guarde os dados no computador.
3. Apresente os dados na tela.

Exemplo do funcionamento

```
RUN
TOTAL INVERNO *******
TOTAL PRIMAVERA ***
TOTAL VERÃO *****
TOTAL OUTONO *****
```

O programa

```
5 LET A = 0
6 LET B = 0
7 LET C = 0
8 LET D = 0
10 FOR I = 1 TO 20
20 PRINT "PESSOA NO."; I; "NASCEU NO"
30 PRINT "INVERNO, PRIMAVERA, VERAO, OUTONO"
40 PRINT "RESPONDA I,P,V,O"
50 INPUT B$
60 IF B$ = "I" THEN LET A = A+1
70 IF B$ = "P" THEN LET B = B+1
80 IF B$ = "V" THEN LET C = C+1
90 IF B$ = "O" THEN LET D = D+1
100 NEXT I
110 PRINT "TOTAL INVERNO";
115 LET N = A
120 GOSUB 200
130 PRINT "TOTAL PRIMAVERA";
135 LET N = B
140 GOSUB 200
150 PRINT "TOTAL VERÃO";
155 LET N = C
160 GOSUB 200
170 PRINT "TOTAL OUTONO";
175 LET N = D
180 GOSUB 200
190 STOP
200 REM: SUB-ROTINA PARA IMPRIMIR
    ASTERISCOS
210 IF N = 0 THEN GOTO 250
220 FOR I = 1 TO N
230 PRINT "*";
240 NEXT I
250 PRINT
260 RETURN
```

Linhas 5 a 8: Variáveis vazias para acumular os totais de cada estação.

Linha 10: "Loop" para repetir a pergunta para cada pessoa.

Linhas 60 a 90: Nas linhas 60 a 90, o computador compara as respostas em B$ e soma 1 à variável correspondente.

Linha 100: Retorna à linha 10.

Linha 120: A sub-rotina faz o computador imprimir uma quantidade de asteriscos igual ao número de cada variável.

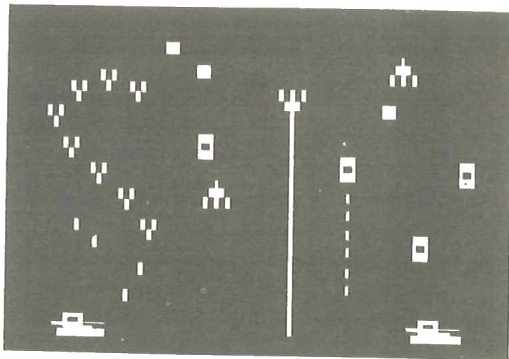
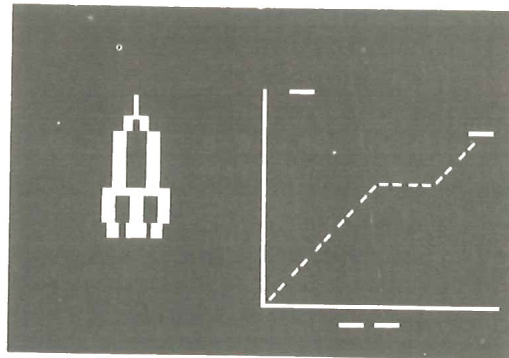
Linhas 135 e 155: Transferindo cada total para N, o programa pode usar a mesma sub-rotina para todas as estações.

Linha 170: O ponto e vírgula faz os asteriscos ficarem na mesma linha.

Linha 210: Esta linha é para o caso de não haver ninguém nascido em uma determinada estação.

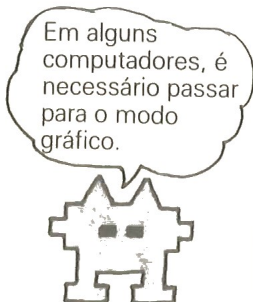
Linha 220: N é igual ao número de asteriscos de cada estação. Este "loop" se repete N vezes, imprimindo na linha 230 um asterisco a cada repetição.

Mais gráficos

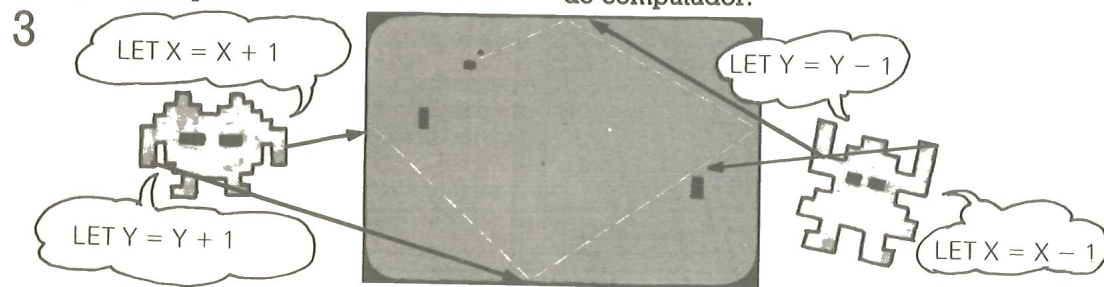
Nestas duas páginas, mostramos como usar PLOT e UNPLOT para criar desenhos animados na tela. Gráficos que se movem são muito úteis para programas de jogos ou para ilustrar na tela os princípios de balística ou gravidade, por exemplo.

Nos jogos eletrônicos ou "videogames", as figuras são controladas por um pequeno computador programado em linguagem de máquina para este fim específico.

Um microcomputador, programado em BASIC, só pode fazer desenhos simples e lentos. Ele não é suficientemente veloz para criar na tela movimentos rápidos.

PLOT/UNPLOT

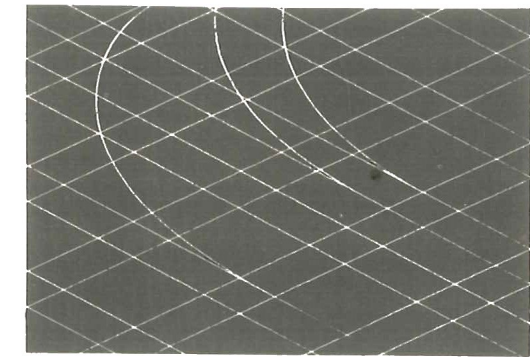
1

```
10 LET X = 1
20 LET Y = 1
30 PLOT (X, Y)
40 UNPLOT (X, Y)
50 LET X = X+1
60 LET Y = Y+1
70 GOTO 30
```

Este pequeno programa faz mover na tela um ponto de luz. Não esqueça que as instruções PLOT e UNPLOT variam em alguns computadores.

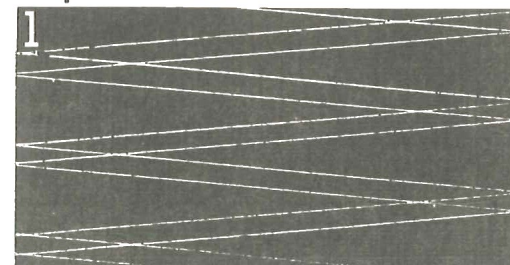
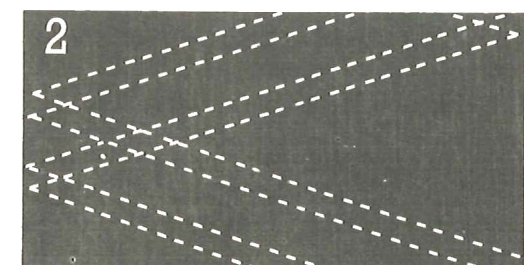
Quando o ponto atinge o limite da tela, o programa é interrompido, já que os valores de X e Y ficaram fora do alcance do computador.

3

Os jogos eletrônicos utilizam programas do tipo daquele que acabamos de ver. São introduzidas algumas regras simples para manter o ponto na tela, evitando a interrupção do programa. Quando o ponto atinge a parte superior da tela, a quantidade que seria somada a Y passa a ser subtraída. Da mesma forma, quando atinge a margem direita, o valor de X passa a ser reduzido, ao invés de aumentado.

Traçado de curvas

Este programa permite traçar na tela uma linha contínua que, atingindo uma das margens do vídeo, toma uma outra direção. Como a instrução UNPLOT não é usada, obtém-se um desenho na tela. A figura à direita dá um exemplo do que aparece no vídeo quando você roda este programa. Na linha 100, o computador é programado para traçar até 10000 pontos. Você pode mudar este número, se quiser, ou interromper a execução do programa a qualquer momento, através do comando BREAK.

```
10 REM: PASSAR PARA O MODO GRAFICO SE NECESSARIO
20 PRINT "DEFINICAO DA TELA, LARGURA";
30 INPUT L
40 PRINT "ALTURA";
50 INPUT H
55 CLS
60 LET X = L/2
70 LET Y = H/2
80 LET S = 1
90 LET T = 1
100 FOR I = 1 TO 10000
110 LET S = S + (INT(RND(1)*10+1)−5)/50
120 LET X = X+S
130 LET Y = Y+T
140 IF X < 5 THEN LET S = −S
150 IF X > L−5 THEN LET S = −S
160 IF Y < 5 THEN LET T = −T
170 IF Y > A−5 THEN LET T = −T
180 GOSUB 300
190 NEXT I
200 STOP
300 REM: PLOTAR LINHA
310 PLOT (X,Y)
320 RETURN
```

- Linhas 20 a 50: As linhas 20 a 50 pedem a largura e a altura da tela em número de pontos.
- Linhas 60–70: Assim, X e Y partem do meio da tela.
- Linhas 80–90: S e T são os valores a somar a X e a Y para que a linha se mova na tela.
- Linha 100: O "loop" das linhas 100 a 190 se repete 10000 vezes e a cada vez X e Y mudam ligeiramente de valor.
- Linha 110: Esta instrução calcula um pequeno valor a somar a X.
- Linhas 140–170: Estas linhas controlam as margens da tela e invertem S e T quando X ou Y chegarem a menos de 5 pontos do limite do vídeo.
- Linha 180: Remete à sub-rotina que imprime o ponto.
- Linha 310: Traça o ponto para as coordenadas X e Y.

Experiências

A linha 110 soma um número aleatório bastante pequeno a X a cada passagem no "loop", o que faz as curvas mudarem de trajetória ligeiramente. Se você tirar esta linha do programa, as curvas passam a ficar paralelas. Experimente mudar os valores de S e T das linhas 80 e 90 para 5 ou 10. Você obterá curvas tracejadas ou pontilhadas.

Programando poemas engraçados

Nas próximas páginas, nós vamos mostrar como fazer um programa capaz de compor dezenas de poesias. Este programa pode ser usado num brinquedo chamado "computador de papel". Nós aqui vamos fazer este programa funcionar num micro de verdade.

Frases

ERA UM JOVEM QUE NASCEU EM
E
O SEU
ERA UM RAPAZ BONITO
QUE SAIU CORRENDO

Palavras

AVARÉ	MACAÉ	GUAPORÉ	ITARARÉ
TIROU	DESLOCOU	MACHUCOU	COLOCOU
DEDO	PESCOÇO	BRAÇO	JOELHO
DO JACARÉ	NO BALÉ	COM O PÉ	NA CHAMIN
MAS DANÇOU	PORÉM DESMAIOU	QUASE REBOLOU	TANTO PULO
NO GRITO	NO APITO	NO RITO	DO MOSQUIT
NO PANGARÉ	DO CHALÉ	DE RÉ	DO BANZÉ

1 A = 0 e B = 0
2 Some 1 a A
3 Se A = 6 vá à linha 10
4 Escreva a linha de dados A
5 Some 1 a B
6 Gire o pião para achar N
7 Escreva os dados da linha B, coluna N
8 Se B = 3 ou 5 vá à linha 5
9 Vá à linha 2
10 Pare

Era um jovem que nasceu em Macaé
E deslocou o seu dedo na chaminé
Era um rapaz bonito
Tanto pulou do mosquito
Que saiu correndo de ré

Aqui está o programa do computador de papel. Ele parece um pouco com o BASIC, mas não funcionaria num computador de verdade. As palavras e frases do poema são "guardadas" em pedaços de papel que o programa seleciona. O pião é um gerador de números aleatórios, entre 1 e 4.

1 Traduzindo o programa para BASIC

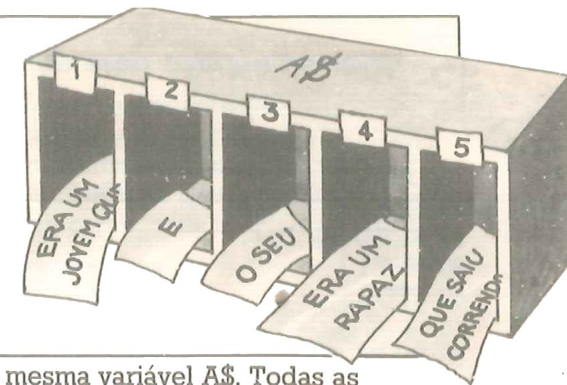
Isto não vai funcionar num computador ainda.

```
10 LET A = 0
20 LET B = 0
30 LET A = A+1
40 IF A = 6 THEN STOP
50 Escrever a frase A
60 LET B = B+1
70 LET N = INT(RND (1)* 4+1)
80 Escrever os dados da linha B coluna N
90 IF B = 2 THEN GOTO 60
100 IF B = 5 THEN GOTO 60
110 GOTO 30
120 END
```

- Linhas 10–20: Estas linhas criam as variáveis.
- Linhas 30–40: As linhas 30 e 40 contam o número de dados selecionados.
- Linha 50: As linhas 50 e 80 ainda não estão em BASIC.
- Linha 60: A linha 60 conta o número de palavras.
- Linha 70: Escolhe um número aleatório de 1 a 4.
- Linhas 90–100: As linhas 90 e 100 fazem retornar para escolher outro dado.

A maior parte do programa é facilmente traduzível para BASIC. Entretanto, as linhas 50 e 80 apresentam problemas. O computador precisa guardar e selecionar as linhas de dados e palavras necessárias para compor cada linha do poema.

2 Entrando com os dados

```
50 READ A$
:
180 DATA ERA UM JOVEM QUE NASCEU
    EM, E, O SEU
190 DATA ERA UM RAPAZ BONITO, QUE
    SAIU CORRENDO
```

Para entrar com as frases e as palavras, use as instruções READ e DATA. Cada vez que o computador encontra a instrução READ, ele apanha um outro item da linha DATA e guarda na variável. É possível guardar todos os dados na mesma variável A$. Todas as variáveis contendo mais de um dado são chamadas de "array" e cada item pode ser identificado por um número. Assim, por exemplo, READ A$ (3) corresponde O SEU.*

3

Uma variável chamada de bidimensional pode guardar os dados em linhas e colunas. Cada dado é indicado pelo número da linha e da coluna. Assim, READ B$ (4,2) significa NO BALÉ e READ B$ (6,3) significa NO RITO. Você também pode guardar números em "arrays", usando uma variável númerica, como N (5,7).

4 Colocando os dados nas variáveis

```
10 FOR I = 1 TO 7  — I é o número da linha
20 FOR J = 1 TO 4  — J é o número da coluna
30 READ B$ (I, J)
40 NEXT J
50 NEXT I
60 DATA AVARE, MACAE, GUAPORE, ITARARE
70 DATA TIROU, DESLOCOU, MACHUCOU, COLOCOU
80 DATA DEDO, PESCOCO, BRACO, JOELHO
90 DATA DO JACARE, NO BALE, COM O PE, NA CHAMINE
100 DATA MAS DANCOU, POREM DESMAIOU, QUASE REBOLOU, TANTO PULOU
110 DATA NO GRITO, NO APITO, NO RITO, DO MOSQUITO
120 DATA NO PANGARE, DO CHALE, DE RE, DO BANZE
```

Para que cada dado passe para a variável, é necessário modificar os números entre parênteses na instrução READ. Um "loop encaixado" pode fazer isto. No exemplo acima, o "loop" do I faz variar o número da linha e o "loop" do J o número da coluna.

* Este programa não funciona nos computadores da família SINCLAIR, ou seus equivalentes no Brasil, como TK82, 83, 85, NE8000 ou CP200. Na página seguinte, damos mais explicações.

5 Criando espaço para as variáveis

```
 5 DIM K$ (5)
10 FOR I = 1 TO 5
20 READ K$ (I)
30 NEXT I
40 STOP

60 DATA GATO, RATO, PULGA, GRILO
```

Linha 5: Este é o tamanho da variável K$, ou seja, 5 itens.

Linha 20: Esta linha coloca dados em K$ cada vez que o "loop" é repetido.

K$ GATO RATO PULGA GRILO

Este programa tem um "grilo". Não tem dados suficientes para K$.

No começo do programa você tem que determinar o tamanho da variável. Para isto, utiliza-se a instrução DIM seguida pelo nome da variável e o número de dados, entre parênteses. Por exemplo, DIM K$(5).

Nas variáveis matriciais bidimensionais, é necessário informar o número de linhas e colunas, por exemplo, DIM C$ (5,3). É necessário informar o número certo de itens, pois do contrário obtém-se um "grilo".

6 Imprimindo os dados

```
200 LET A = 0
210 LET B = 0
220 LET A = A + 1
230 IF A = 6 THEN STOP
240 PRINT A$ (A)
250 LET B = B+1
260 LET N = INT(RND(1)*4+1)
270 PRINT B$ (B,N)
280 IF B = 3 THEN GOTO 250
290 IF B = 5 THEN GOTO 250
300 GOTO 220
310 END
```

Linha 220: Conta quantas vezes esta parte do programa é repetida.

Linha 250: B conta as linhas da matriz B$ e certifica-se de que a linha correta é usada.

As linhas 280 e 290 fazem, quando B for igual a 3 ou 5, o computador imprimir mais uma palavra selecionada da matriz B$ antes de imprimir uma frase guardada na variável A$.

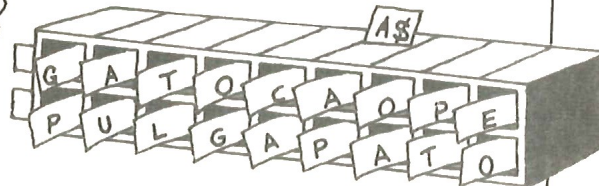
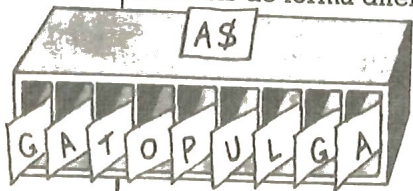
Linha 300: Faz o computador imprimir a próxima frase guardada em A$.

Este programa faz o computador imprimir as frases e as palavras na ordem correta. Esta seção do programa é repetida 5 vezes. Cada vez, o computador imprime a frase de número A e alguma palavra da matriz B$ da linha B. A palavra selecionada será decidida pela variável aleatória N, que pode ser 1, 2, 3 ou 4.

Os computadores da linha SINCLAIR, TK, CP200 e as variáveis

Este programa não funciona, como está, nos computadores da família TK, NE8000, CP200, Sinclair e equivalentes, porque eles tratam as variáveis de forma diferente.

Em matrizes bidimensionais, é necessário indicar o número da linha, assim como os números dos caracteres. Por exemplo, A$ (2, 6 TO 9) é PATO.

A$ G A T O P U L G A

Para fazer um computador desta linha selecionar um dado em particular, é necessário fornecer a posição do primeiro e último caractere do item desejado. É o mesmo método usado por estes microcomputadores para LEFT $, RIGHT $ etc. (Ver página 32.)

No começo do programa, indica-se quantas linhas a matriz vai ter e quantos caracteres por linha. Por exemplo: DIM A$ (2,9) significa duas linhas, cada uma com 9 caracteres. Cada linha da matriz tem que ter o mesmo número de caracteres.

O programa completo de poemas engraçados

Agora podemos juntar as partes do programa de poemas engraçados. A primeira parte do programa (linhas 10 a 190) alimenta o computador com os dados e a segunda parte (linhas 200 a 310) imprime o poema. Cada vez que você rodar o programa, aparecerá uma versão diferente da poesia, já que o número aleatório N faz o computador selecionar palavras diferentes em B$.

```
 10 DIM A$ (5)
 20 DIM B$ (7,4)
 30 FOR I = 1 TO 7
 40 FOR J = 1 TO 4
 50 READ B$ (I, J)
 60 NEXT J
 70 NEXT I
 80 DATA AVARE, MACAE, GUAPORE, ITARARE
 90 DATA TIROU, DESLOCOU, MACHUCOU, COLOCOU
100 DATA DEDO, PESCOCO, BRACO, JOELHO
110 DATA DO JACARE, NO BALE, COM O PE, NA CHAMINE
120 DATA MAS DANCOU, POREM DESMAIOU, QUASE REBOLOU, TANTO PULOU
130 DATA NO GRITO, NO APITO, NO RITO, DO MOSQUITO
140 DATA NO PANGARE, DO CHALE, DE RE, DO BANZE
150 FOR I = 1 TO 5
160 READ A$ (I)
170 NEXT I
180 DATA ERA UM JOVEM QUE NASCEU EM, E, O SEU
190 DATA ERA UM RAPAZ BONITO, QUE SAIU CORRENDO
200 LET A = 0
210 LET B = 0
220 LET A = A+1
230 IF A = 6 THEN STOP
240 PRINT A$ (A)
250 LET B = B+1
260 LET N = INT(RND(1)*4+1)
270 PRINT B$ (B,N)
280 IF B = 3 THEN GOTO 250
290 IF B = 5 THEN GOTO 250
300 GOTO 220
310 END
```

As linhas 10 e 20 reservam espaço para as variáveis matriciais — uma linha de 5 itens para A$ e 7 linhas de 4 itens para B$.

Linhas 30 a 70: "Loops encaixados" para alimentar os dados em B$.

As linhas 80 e 140 contêm todas as palavras a serem guardadas em B$.

Linhas 150 a 170: "Loop" para colocar os dados em A$.

As linhas 180 e 190 contêm todos os dados para guardar em A$.

Linha 240: Faz imprimir a frase guardada em A$ no compartimento A.

Linha 270: Faz imprimir as palavras guardadas em B$, linha B, coluna N.

O programa pára na linha 230 quando A = 6, e na realidade nunca atinge a linha 310, mas em alguns computadores é necessário um END em todos os programas.

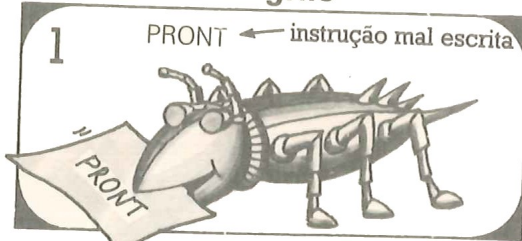
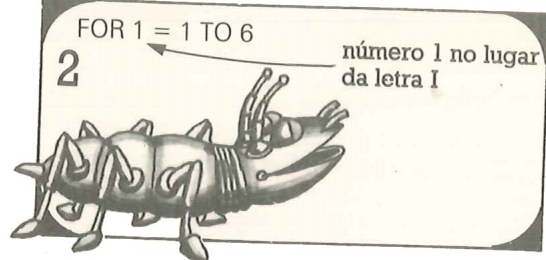
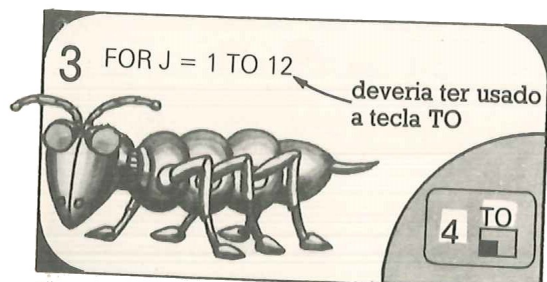
Exemplos

ERA UM JOVEM QUE NASCEU EM	ERA UM JOVEM QUE NASCEU EM
AVARE	ITARARE
E	E
DESLOCOU	MACHUCOU
O SEU	O SEU
DEDO	PESCOCO
NA CHAMINE	COM O PE
ERA UM RAPAZ BONITO	ERA UM RAPAZ BONITO
MAS DANCOU	QUASE REBOLOU
DO MOSQUITO	NO APITO
QUE SAIU CORRENDO	QUE SAIU CORRENDO
DO BANZE	DE RE

Aqui estão duas das 16384 versões possíveis do poema. Se, ao tentar rodar este programa, você sempre obtiver o mesmo poema, leia o seu manual de instruções, já que certos computadores produzem a mesma seqüência de números aleatórios cada vez que são ligados.

Dicas de programação

Nestas duas páginas, você encontra algumas "dicas" para ajudá-lo a programar, uma lista dos "grilos" mais comuns e o que possa tê-los causado. Se você tiver um programa que não funcione, procure na lista abaixo o possível "grilo".

Procurando o "grilo"

Procure erros de datilografia nas instruções de BASIC. Uma instrução mal escrita não será reconhecida pelo computador.

Verifique os ós e os zeros, os Is e os algarismos 1, para se certificar de que não houve troca de um pelo outro.

Se o seu computador for da família TK, ou CP200, verifique se você não terá entrado a instrução letra a letra, ao invés de usar a tecla correspondente.

Programando

Quando você estiver programando, é necessário lembrar que o computador executa três tipos diferentes de tarefas: instruções simples, repetições e tomadas de decisão. Estas são as bases de todos os programas.

```
LET A = 3
LET N = N + 1
PRINT A/T
PLOT (X,Y)
```

```
FOR J = 1 TO 6
20 LET A = 1
20 IF A < 10 THEN
   GOTO 100
```

TOMADAS DE DECISÃO

```
IF X = Y THEN STOP
IF K$ = "ALO"
THEN PRINT A
```

Este livro cobriu todas as principais instruções em BASIC necessárias para o computador executar estas tarefas. Ao programar, determine as necessidades de cada etapa, e então decida que instruções usar.

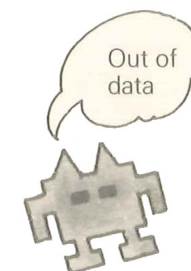
Verifique se você não esqueceu nenhuma aspa ou vírgula, sobretudo nas instruções mais complexas.

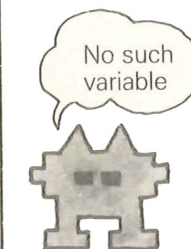
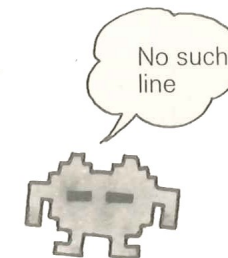
Existem diversas maneiras de escrever um mesmo programa. Algumas são mais claras e rápidas do que outras. Ao fazer um programa longo, é uma boa idéia dividi-lo em várias partes independentes, através de sub-rotinas. A parte principal do programa pode ser um grupo compacto de instruções, decisões e repetições que controlam as sub-rotinas.

Dividindo o programa em seções como esta, você o torna mais fácil de entender e testar. Podem-se procurar os eventuais erros, etapa por etapa, sem ter que rodar todo o programa. Lembre-se, entretanto, de colocar um nome para cada seção, através da instrução REM.

Certifique-se do uso correto das instruções RND, PLOT e CLS no seu computador, já que elas variam conforme o fabricante. Observe também se o seu computador precisa passar para o modo gráfico antes de aceitar o comando PLOT. Não esqueça que os números decimais usam ponto ao invés da vírgula.

Mensagens erradas

Todos os computadores imprimem uma mensagem quando há um "grilo" no programa. As mensagens estão explicadas no manual do computador. Aqui estão as mais comuns:

◀ Significa que não há itens suficientes para o computador ler nas linhas DATA. Pode ser que você tenha omitido uma vírgula entre dois itens, o que fez o computador assumi-los como sendo um só.

▶ GOTO ou GOSUB remetem o computador a uma linha inexistente. Você pode ter acidentalmente apagado esta linha ou trocado um algarismo ao dar-lhe entrada.

◀ Esta mensagem aparece no TK ou CP200 e indica que você não iniciou a variável antes de utilizá-la. Inicia-se a variável escrevendo LET C = 0 ou LET C$ = " "

▶ Significa que falta a instrução NEXT que encerra o "loop". Pode ser que você tenha trocado o nome da variável ou ao invés da letra I tenha colocado o algarismo 1.

Uma última palavra

Certos "grilos" são muito difíceis de localizar, mas se o programa não funciona é porque há um "grilo" em algum lugar. Se não conseguir encontrá-lo, tente bater de novo as linhas mais complicadas. Pode ser que assim você corrija o erro e o programa funcione.

Respostas dos quebra-cabeças

Página 15
Programa do nome e mensagem

```
10 PRINT "QUAL E O SEU NOME?"
20 INPUT N$
30 PRINT "BOM DIA"
40 PRINT N$
50 PRINT "COMO VAI?"
```

Página 17
1. Programa das somas

```
10 LET A = 9
20 LET B = 7
30 PRINT A * B
40 PRINT A/B
50 LET A = A + 1
60 LET B = B + 3
70 PRINT A * B, A/B
80 END
```

vírgula para deixar espaço

2. Programa da tabuada

espaços

```
30 PRINT A; " VEZES "; B " E IGUAL A "; A*B
40 PRINT A; " DIVIDIDO POR "; B; " E IGUAL
   A "; A/B
```

3. Alterações no programa de nome e mensagem

```
10 PRINT "QUAL E O SEU NOME?"
20 INPUT N$
30 PRINT "BOM DIA"; N$; "COM0 VAI?"
```

Página 18
Programa da tabuada

```
10 PRINT "QUANTO E 7 VEZES 7"
20 INPUT A
30 IF A = 49 THEN PRINT "CERTO"
40 IF A <> 49 THEN PRINT "NAO"; 7 * 7
```

não esqueça este ponto e vírgula

Página 19
Alterações no programa "Qual é a sua idade"

Substitua a linha 30 e introduza a linha 35 abaixo:

```
30 IF G < 14 THEN PRINT "MAIS VELHO
   QUE ISTO"
35 IF G > 14 THEN PRINT "MAIS JOVEM
   QUE ISTO"
```

Página 23
Um contador para plotar

```
 5 LET C = 0
45 LET C= C + 1
50 IF C < 6 THEN GOTO 10
```

Plotando a sua inicial

Aqui está um exemplo para plotar a letra L

```
10 LET X = 15
20 LET Y = 30
30 PLOT (X,Y)
40 LET Y = Y - 1
50 IF Y > 5 THEN GOTO 30
60 LET X = X + 1
70 PLOT (X,Y)
80 IF X < 45 THEN GOTO 60
90 END
```

Página 24
Números aleatórios

A fórmula para selecionar um número aleatório entre 10 e 20 é INT (RND(1)*11 + 9). Nos computadores que precisam de somente um número entre parênteses depois do RND, a fórmula seria RND(11)+9. Existem onze números entre 10 e 20; assim, basta você selecionar um número entre 1 e 11 e somar 9.

Página 25
Ataque espacial

Aqui estão as instruções necessárias para adicionar um contador dos tiros certos:

```
15 LET S = 0
75 IF X = A*B THEN LET S = S + 1
95 PRINT "VOCE ACERTOU "; S; " EM 6
   EXTRATERRESTRES "
```

Página 27
1. Tabuada de 8

```
10 PRINT "TABUADA DE MULTIPLICAR
   POR 8"
20 FOR J = 1 TO 12
30 PRINT J; "X8 ="; J*8
40 NEXT J
```


Página 27
2. Tabuada de N

```
10 INPUT "ENTRE UM NUMERO"; N
20 PRINT "AQUI ESTA A TABUADA DE
   MULTIPLICAR POR "; N
30 FOR I = 1 TO 12
40 PRINT I; " VEZES "; N; " E IGUAL A "; I*N
50 NEXT I
60 INPUT "UM OUTRO NUMERO (SIM OU
   NAO)"; M$
70 IF M$ = "SIM" THEN GOTO 10
```

Nos computadores TK82/83, NE 8000 ou CP200 é necessário incluir uma instrução PRINT separada do INPUT.

Página 32
Quebra-cabeça "Livro de computadores"

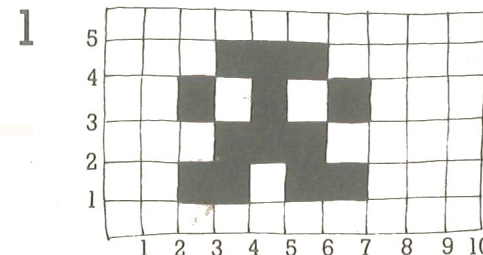
LEFT$ (A$,8) é "LIVRO DE"
RIGHT$ (A$,10) é "COMPUTADORES"
MID$ (A$,7,7) é "DE COMP"

Página 34
Programa do jogo numérico

```
10 PRINT "PENSE NUM NUMERO"
20 PRINT "MULTIPLIQUE-O POR 2 E
   SOME 4"
30 PRINT "DIVIDA-O POR 2 E SOME 7"
40 PRINT "MULTIPLIQUE-O POR 8, SUB-
   TRAIA 12"
50 PRINT "DIVIDA-O POR 4 E SUBTRAIA
   11"
60 PRINT "DIGA-ME O RESULTADO"
70 INPUT N
80 PRINT "O NUMERO QUE VOCE PEN-
   SOU ERA"; (N-4)/2
```

Os parênteses são necessários para que a subtração seja feita antes da divisão.

Programa para vários extraterrestres

1

Desenhe um extraterrestre usando uma folha de papel quadriculado.

2

4,5; 5,5; 6,5;
3,4; 5,4; 7,4;
4,3; 5,3; 6,3;
3,2; 4,2; 6,2; 7,2

Anote as coordenadas de cada quadrado que compõe o invasor extraterrestre.

3

```
5 CLS
50 INPUT "QUANTOS PONTOS NA LARGURA DA
   TELA"; L
60 INPUT "QUANTOS PONTOS
   NA ALTURA"; H
65 CLS
70 FOR I = 0 TO H STEP H/6
80 FOR J = 0 TO L STEP L/6

130 NEXT J
140 NEXT I
150 END
```

Mude 6 para um número maior se quiser aumentar o número de extraterrestres que aparecem na tela. (Se você arrumar um "grilo" é porque escolheu um número muito grande)

Coloque as linhas de PLOT aqui. Por exemplo:
90 PLOT (J+3, I+2)
92 PLOT (J+4, I+2)
são as linhas necessárias para desenhar a perna esquerda do ET conforme o desenho acima. Você precisa de uma linha de programa para cada quadrado.

Copie o programa de Repetição de Desenhos da página 29, eliminando as linhas 10 a 40 e 90 a 120 como mostramos acima. (Estas linhas geram desenhos aleatórios e você não precisa delas.)
Agora você precisa introduzir as instruções de PLOT no programa, entre as linhas 80 e 120. Para cada par de coordenadas, é necessário somar J ao primeiro número e I ao segundo, para fazer o ser extraterrestre se repetir na tela.

Pequeno dicionário de BASIC

Aqui está a lista das expressões BASIC utilizadas neste livro, com uma pequena explicação do que elas significam. Algumas das instruções, como CLS, não são padronizadas em todos os computadores, e para diferenciá-las colocamos um asterisco antes de cada uma. Verifique o manual do seu computador.

* BREAK: Em alguns computadores esta instrução interrompe o programa. Entretanto, tenha cuidado, pois em outros ela apaga da memória todo o programa. Neste caso, você deve usar a instrução ESCAPE ou qualquer outra correspondente.
* CLS: Limpa a tela.
* DATA: Uma lista de itens, por exemplo, palavras ou números, que serão guardados em variáveis. Ver READ.

DIM: Indica ao computador o espaço a ser reservado para uma variável. Por exemplo: DIM A$ (5,4) significa que a variável ocupará cinco linhas de quatro colunas.

* EDIT: Permite modificar uma linha de um programa sem ter que rebatê-la toda de novo.
* END: Indica ao computador o fim do programa. Alguns modelos necessitam desta instrução; outros, como os da família Sinclair (TK 82, 83, 85, NE-8000 ou CP-200), podem passar sem ela.

FOR... NEXT: Cria um "loop" no programa, fazendo repetir as instruções nele contidas por um determinado número de vezes.

GOSUB: Faz o computador sair do corpo principal do programa e executar uma sub-rotina.

GOTO: Remete o computador a uma outra linha do programa.

IF... THEN: Compara dados (por exemplo, números, palavras ou conteúdo de variáveis) e age em função dos resultados.

INPUT: Faz o computador pedir um dado no decorrer do programa.

INT: Transforma um número decimal em um número inteiro, ignorando os algarismos à direita do ponto.
Exemplo: INT (3.4) =3. Os números decimais em BASIC, ao invés da vírgula, usam ponto.

* LEFT$: Indica ao computador para tomar um certo número de caracteres situados do lado esquerdo de uma cadeia alfanumérica. LEFT$ (A$,4) significa tomar quatro caracteres da esquerda de A$.

LEN: Informa o tamanho de uma cadeia, isto é, o número de caracteres da variável.

LET: Dá um nome a uma variável e coloca nela uma informação. Exemplo: LET N = 4 ou LET B$ = "GATO".

* LIST: Imprime o programa na tela.
* MID$: Indica ao computador para selecionar caracteres do meio de uma cadeia alfanumérica. Exemplo: MID$ (A$, 4,3) significa selecionar três letras começando na quarta letra de A$.

NEW: Limpa o programa da memória.

* NEWLINE (tecla): Indica ao computador que acabamos de entrar uma instrução ou um dado. Em alguns equipamentos, esta tecla chama-se RETURN ou ENTER.

NEXT: Veja FOR.

* PLOT: Faz acender um ponto na tela: PLOT (X,Y) acende o ponto de coordenadas X e Y.

PRINT: Faz imprimir alguma coisa na tela.

* READ: Faz o computador ler a informação na linha DATA e guardá-la numa variável. Ver DATA.
* READY: Mensagem dada por alguns computadores quando estão prontos para receber uma nova instrução.

RETURN: É usada no fim de uma sub-rotina e remete o computador à instrução depois de GOSUB. Ver GOSUB.

REM: O computador ignora a linha iniciada com REM, mas a imprime quando o programa é listado. Bastante útil para relembrar o que cada parte do programa faz.

* RIGHT$: Indica ao computador para selecionar os caracteres à direita da cadeia. Exemplo: RIGHT$ (A$, 4) significa selecionar os quatro caracteres contados a partir da direita de A$.
* RND: Seleciona um número aleatório.

RUN: Faz o computador executar o programa.

SQR: Faz o computador extrair a raiz quadrada de um número.

STEP: Usado junto com a instrução FOR... NEXT. Indica quando o computador deve repetir o "loop".

STOP: Usado no programa para interromper a execução.

THEN: Veja IF.

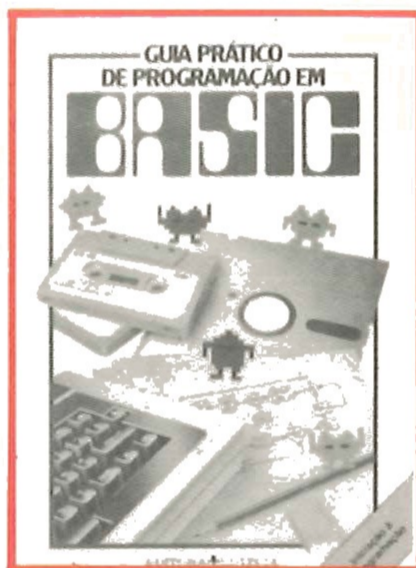
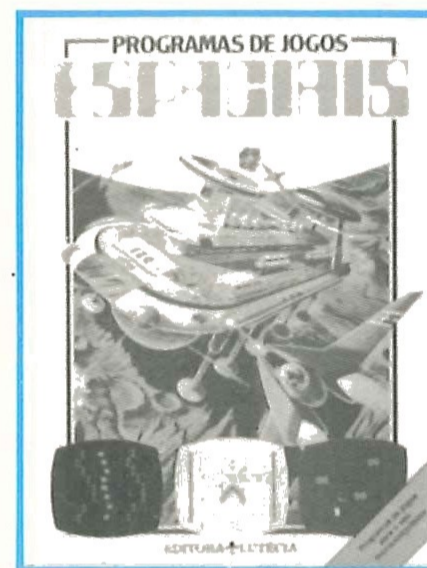
* UNPLOT: Faz o computador apagar um ponto na tela. O contrário de PLOT.

Glossário

Array: Ver MATRIZ.
Bug: O mesmo que "grilo".
Cadeia: Uma série de caracteres para serem guardados numa variável (ex: ABC123)
CPU: Ver UCP.
Cursor: Um sinal luminoso na tela (um ponto, quadrado, letra ou símbolo) que indica onde o próximo caractere será impresso.
Erro de sintaxe: Um erro de ortografia nas instruções de BASIC.
Fluxograma: Um gráfico das principais operações do programa, útil para planejar a programação.
Gráfico: Informação apresentada visualmente na tela.
"Grilo": Um erro no programa.
K: A. unidade de medida do tamanho da memória de um computador. Um K é igual a 1024 "bytes", ou seja, 1024 caracteres, já que na maioria dos micros um caractere ocupa um "byte".
Matriz: Um conjunto de variáveis contendo diversos dados ordenados.
Pixel: Um pequeno quadrado na tela que o computador acende e apaga para fazer gráficos ou desenhos.
Programa: Uma lista de instruções indicando ao computador como executar determinada tarefa.
RAM: Memória temporária de um computador. O nome foi formado com as iniciais de Random Access Memory (Memória de Acesso Aleatório). Na memória RAM são guardados os dados de um programa e os resultados obtidos pelo computador.
ROM: Memória permanente de um computador. O nome foi formado com as iniciais Read Only Memory (Memória Apenas para Ler). Na memória ROM são guardadas as instruções para o funcionamento geral do computador.
Sub-rotina: Uma parte do programa destinada a executar uma determinada tarefa, a qual é geralmente repetida diversas vezes durante o desenrolar do programa.
UCP — Unidade Central de Processamento — Parte do computador onde todas as operações são controladas, como por exemplo as comparações, somas etc.
Variável: Um espaço de memória identificado por um nome (etiqueta ou "label") e que contém informações.

Guias Práticos de Microcomputadores

O que é capaz de fazer um microcomputador! Calcular, evidentemente, mas também organizar perguntas, escrever poemas, jogar uma quantidade de jogos cada qual mais palpitante, até mesmo compor música... Pequenos guias práticos de introdução à microinformática, as obras desta nova coleção nos permitem descobrir todas as possibilidades que os microcomputadores nos oferecem. Eles nos iniciam na linguagem e no funcionamento do computador, na aprendizagem da programação e — por que não? — na criação de programas originais: a clareza dos textos, a alegria das cores, a graça dos desenhos, tudo é concebido nestes livros para fazer desta iniciação um prazer.

Um colorido guia para melhor compreendermos os microcomputadores — como trabalham e o que fazem, apresentando idéias de coisas que podem ser feitas com o micro. Apresenta um utilíssimo guia comparativo dos diversos micros existentes no mercado brasileiro.

Um guia passo a passo de programação para os absolutamente iniciantes. Apresenta as diferentes linguagens e aprofunda no BASIC, incluindo pequenos programas, simples e divertidos. Profusamente ilustrado a cores.

13 programas fascinantes e ao mesmo tempo simples, para todo tipo de microcomputadores; Jogos Intergaláticos, Módulo Lunar, Viagem ao Futuro, Salvamento no Espaço e muitos outros, com respostas dos problemas.

Introduz o leitor na programação BASIC, através de exemplos didáticos, e é sobretudo indicado ao jovem que começa a programar. Substitui com vantagens os manuais dos fabricantes. Apresenta dezenas de dicas utilíssimas para aprimorar a técnica de programação.

13 listagens de programas de jogos, para serem "rodados" nos micros pessoais mais difundidos no Brasil. Muito útil para ensinar o leitor a desenvolver seus próprios programas. Além disso, é um passatempo fascinante.

Um verdadeiro caderno a cores de problemas e exercícios, com jogos e quebra-cabeças , ideais para todo curso de programação em BASIC. Todas as respostas e explicações estão incluídas, para ajudar a esclarecer todo e qualquer problema.